DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93943

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A COMPUTER PROGRAM TO MODEL PASSIVE ACOUSTIC ANTISUBMARINE SEARCH USING MONTE CARLO SIMULATION TECHNIQUES

by

Steven Gregory Slaton

September, 1983

Thesis Advisor: J. N. Eagle

Approved for public release; distribution unlimited

T215696

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
A Computer Program to Model Passive Acoustic Antisubmarine Search Using Monte Carlo Simulation Techniques

5. TYPE OF REPORT & PERIOD COVERED
Master's Thesis
September, 1983

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Steven Gregory Slaton

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93943

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93943

12. REPORT DATE
September, 1983

13. NUMBER OF PAGES
242

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Program	Passive Sonar
Simulation	Search
Passive Acoustics	Detection

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer program (written in FORTRAN) is presented which uses Monte Carlo techniques to simulate one-searcher, one-target passive acoustic ASW search that terminates at detection. A threshold crossing detection model is used, and stochastic variations in the acoustic signal are modeled using either a Lambda-Sigma Jump or Gauss-Markov error process. Both platforms have the capability of detecting each other, and area and

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 Continued:

barrier searches are modeled. Features of the program include interactive data input, extensive use of graphical displays, and thorough statistical analysis of the results of the simulation.

Approved for public release; distribution unlimited

A Computer Program to Model Passive Acoustic Antisubmarine Search Using Monte Carlo Simulation Techniques

by

Steven Gregory Slaton
Lieutenant Commander, United States Navy
B.S., University of California, Berkeley, 1970

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September, 1983

[illegible]

ABSTRACT

A computer program (written in FORTRAN) is presented which uses Monte Carlo techniques to simulate one-searcher, one-target passive acoustic ASW search that terminates at detection. A threshold crossing detection model is used, and stochastic variations in the acoustic signal are modeled using either a Lambda-Sigma Jump or Gauss-Markov error process. Both platforms have the capability of detecting each other, and area and barrier searches are modeled. Features of the program include interactive data input, extensive use of graphical displays, and thorough statistical analysis of the results of the simulation.

TABLE OF CONTENTS

I. DESCRIPTION OF THE SIMULATION MODEL -------------- 14

- A. INTRODUCTION ---------------------------------- 14
- B. GENERAL PROGRAM DESCRIPTION ------------------ 16
- C. SEARCH SCENARIOS USED IN PASS --------------- 17
  - 1. Area Search ------------------------------ 17
  - 2. Barrier Search --------------------------- 17
- D. THE ENVIRONMENTAL MODEL --------------------- 18
- E. THE DETECTION MODEL ------------------------- 18
  - 1. Searcher Figure-of-Merit ---------------- 19
  - 2. Target Figure-of-Merit ------------------ 20
  - 3. The Threshold Crossing Model With a Stochastic Error Process ---------------- 20
- F. STOCHASTIC ERROR PROCESSES ------------------ 22
  - 1. The Lambda-Sigma Jump Error Process ----- 23
  - 2. The Gauss-Markov Error Process ---------- 25
- G. A THREE-OUT-OF-FIVE DETECTION CRITERIA MODEL ------------------------------------- 27
- H. A SEARCH MEASURE OF EFFECTIVENESS (MOE) ----- 28
  - 1. Searcher MOE ---------------------------- 28
  - 2. Target MOE ------------------------------ 29
  - 3. The Exchange Ratio ---------------------- 30
- I. PROGRAM RESULTS/OUTPUT AND STATISTICAL ANALYSIS ----------------------------------- 30
- J. VALIDATION OF THE PROGRAM ------------------ 31
- K. PROGRAM RUN TIME ---------------------------- 32

II. DESCRIPTION OF PROGRAM PASS ---------------------- 33

A. INTRODUCTION ---------------------------------- 33

B. DATA INPUT AND SIMULATION INITIALIZATION (PROGRAM PARTS 1,2,3) ------------------------ 33

C. REPLICATION COUNTER AND SETUP (PROGRAM PART 4) ------------------------------ 35

D. INITIALIZATION OF THE REPLICATION (PROGRAM PART 5) ------------------------------ 35

E. DETERMINE DETECTION RANGES (PROGRAM PART 6) ------------------------------ 36

F. CHECK FOR DETECTION (PROGRAM PART 7) -------- 37

G. COMPUTE THE TIME OF THE "NEXT EVENT" (PROGRAM PART 8) ------------------------------ 38

H. MOVE THE SHIPS (PROGRAM PARTS 9A, 9B) ------- 40

I. BRANCH TO THE "NEXT EVENT" (PROGRAM PART 10) ----------------------------- 41

J. ACOUSTIC FLUCTION LEVEL CHANGE (PROGRAM PART 11) ----------------------------- 42

K. SEARCHER COURSE CHANGE (PROGRAM PART 12) ---- 42

L. TARGET COURSE CHANGE (PROGRAM PART 13) ------ 43

M. SEARCHER SPEED CHANGE (PROGRAM PART 14) ----- 44

N. TARGET SPEED CHANGE (PROGRAM PART 15) ------- 44

O. DETERMINE WHICH PLATFORM CAN DETECT (PROGRAM PART 16A) ---------------------------- 44

P. SIMULTANEOUS DETECTION OPPORTUNITY (PROGRAM PART 16B) ---------------------------- 45

Q. SEARCHER SECURE DETECTION OPPORTUNITY (PROGRAM PART 16C) ---------------------------- 46

R. TARGET SECURE DETECTION OPPORTUNITY (PROGRAM PART 16D) ---------------------------- 46

S. NO DETECTION OCCURS (PROGRAM PART 17) ------- 47

T. RESULTS OUTPUT (PROGRAM PART 18) ------------ 47

III. EXAMPLE SEARCH PROBLEMS ------------------------- 51

A. INTRODUCTION ------------------------------ 51

B. AREA SEARCH: NOISE QUIETING VS. IMPROVED SENSOR PERFORMANCE -------------------------- 52

1. Scenario Description -------------------- 52

2. PASS Input Data ------------------------- 52

3. Environment and Search Plan ------------- 52

4. Platform Parameters --------------------- 52

5. Simulation Run Parameters --------------- 53

6. Results and Conclusions ----------------- 53

C. BARRIER SEARCH: SOME FACTORS AFFECTING PENETRATION SUCCESS ------------------------ 74

1. Scenario Description -------------------- 74

2. PASS Input Data ------------------------- 74

3. Environment and Search Plan ------------- 74

4. Platform Parameters --------------------- 75

5. Results and Conclusions ----------------- 75

D. DEFINITE RANGE LAW APPROXIMATIONS ----------- 101

1. Area Search: Best Patrol Speed to Evade a Systematic Searcher ------------------- 101

E. CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK ------------------------------ 118

APPENDIX A: PASS USERS GUIDE ------------------------- 119

A. INTRODUCTION ------------------------------ 119

1. Data File Management -------------------- 119

2. Required Input Data --------------------- 120

3. The Executive Program ------------------- 121

B. PROGRAM PASS EXECUTION ---------------------- 122

1. Starting the Program: Invoking the Executive Program ----------------------- 122
2. OPTION NO. 1: Terminate or Proceed ------ 122
3. MASTER OPTION: Accept Program Defaults -- 122
4. OPTION NO. 2A: Signal Integration Model - 123
5. OPTION NO. 2: Acoustic Fluctuation Model ---------------------------------- 123
6. OPTION NO. 3A: Existence of File PASS DATA ----------------------------------- 123
7. OPTION NO. 3: Search Area Size ---------- 123
8. OPTION NO. 4: Searcher Track ----------- 124
9. OPTION No. 5: Searcher Direct Path Propagation Loss ------------------------ 125
10. OPTION NO. 6: Target Direct Path Propagation Loss ------------------------ 126
11. OPTION NO. 7: Searcher Convergence Zones ---------------------------------- 127
12. OPTION NO. 8: Target Convergence Zones -- 128
13. OPTION NO. 9: Area or Barrier Search ---- 128
14. OPTION NO. 9A: Initial Target Lateral Distribution --------------------------- 129
15. OPTION NO. 10A: Searcher FOM Dependent on Target Aspect ---------------------- 129
16. OPTION NO. 10: Searcher Figure-of-Merit - 130
17. OPTION NO. 11: Searcher Sprint and Drift Speeds --------------------------- 130
18. OPTION NO. 12: Searcher Sprint and Drift Times ---------------------------- 131
19. OPTION NO. 13: Target Figure-of-Merit --- 132
20. OPTION NO. 14: Target Speed Range ------- 133

21. OPTION NO. 15: Rate of Target Zigs ------ 133

22. OPTION NO. 16: Acoustic Fluctuation Rate Parameters -------------------------- 134

23. OPTION NO. 17: Acoustic Fluctuation Scale Parameters ------------------------- 135

24. OPTION NO. 18: Run Parameters ----------- 136

25. OPTION NO. 19: Input Data to Output File ------------------------------------ 137

26. OPTION NO. 20: Statistical Analysis ----- 137

C. NORMAL RUNNING INDICATIONS ------------------- 137

D. GRAPHICAL PRESENTATION OF RESULTS: PROGRAM PASPLT -- 138

APPENDIX B: DESCRIPTION OF VARIABLE NAMES USED IN PASS ------------------------------------ 139

APPENDIX C: ABBREVIATIONS AND ACRONYMS -------------- 151

APPENDIX D: APPLICATION OF THE PASSIVE SONAR EQUATION TO THE PASS MODEL ------------------------ 154

APPENDIX E: COMPUTER SIMULATION OF THE STOCHASTIC ERROR PROCESSES -------------------------- 156

A. THE LAMBDA-SIGMA JUMP PROCESS ---------------- 156

B. THE GAUSS-MARKOV PROCESS --------------------- 158

C. GENERATION OF PSUEDORANDOM NUMBERS ----------- 160

D. CORRELATION OF THE ERROR PROCESSES ----------- 160

APPENDIX F: A THREE-OUT-OF-FIVE DETECTION MODEL ------ 162

APPENDIX G: CODE FOR EXECUTIVE PROGRAM TO RUN PASS --- 165

APPENDIX H: FORTRAN CODE FOR PASS ------------------- 167

APPENDIX I: FORTRAN CODE FOR PASPLT ----------------- 233

LIST OF REFERENCES ----------------------------------- 240

INITIAL DISTRIBUTION LIST ---------------------------- 242

LIST OF TABLES

I. INPUT DATA FOR CASE A-I ------------------------ 55

II. NUMERICAL RESULTS FOR CASE A EXAMPLES ---------- 56

III. INPUT DATA FOR CASE B-I ------------------------ 77

IV. NUMERICAL RESULTS FOR CASE B EXAMPLES ---------- 78

V. SEARCHER FOM VS. TARGET ASPECT ----------------- 78

VI. INPUT DATA FOR CASE C-I ------------------------ 104

VII. NUMERICAL RESULTS FOR CASES C-I AND C-II ------- 105

VIII. MSEL MAXIMUM EXPECTED INTEGRATION TIMES -------- 164

LIST OF FIGURES

2.1 Flow Diagram for Program PASS Parts 1-7 --------- 48

2.2 Flow Diagram for Program PASS Parts 8-15 -------- 49

2.3 Flow Diagram for Program PASS Part 16 ----------- 50

3.1 Searcher Propagation Loss Curve for Case A Examples -- 57

3.2 Target Propagation Loss Curve for Case A Examples -- 58

3.3 Searcher Track for Case A Examples -------------- 59

3.4 Typical Figures-of-Merit for A-I ---------------- 60

3.5 Cumulative Probability of Detection for A-I ----- 61

3.6 Probability of Detection Vs. Range for A-I ------ 62

3.7 Searcher Position at Time of Detection for A-I -- 63

3.8 Target Position at Time of Detection for A-I ---- 64

3.9 Typical Figure-of-Merit for A-III --------------- 65

3.10 Cumulative Probability of Detection for A-III --- 66

3.11 Cumulative Probability of Counter-Detection for A-III --------------------------------------- 67

3.12 Probability of Detection Vs. Range for A-III ---- 68

3.13 Searcher Position at Time of Detection for A-III -- 69

3.14 Target Position at Time of Detection for A-III -- 70

3.15 Probability of Counter-Detection Vs. Range for A-III --------------------------------------- 71

3.16 Target Position at Time of Counter-Detection for A-III --------------------------------------- 72

3.17 Searcher Position at Time of Counter-Detection for A-III --------------------------------------- 73

3.18 Searcher Propagation Loss for Case B Examples --- 79

3.19 Target Propagation Loss for Case B Examples ----- 80

3.20 Searcher Track for Case B Examples --------------- 81

3.21 Typical Figure-of-Merit for B-I ----------------- 82

3.22 Cumulative Probability of Detection for B-I ----- 83

3.23 Probability of Detection Vs. Range for B-I ------ 84

3.24 Searcher Position at Time of Detection for B-I -- 85

3.25 Target Position at Time of Detection for B-I ---- 86

3.26 Typical Figure-of-Merit for B-II ---------------- 87

3.27 Cumulative Probability of Detection for B-II ---- 88

3.28 Probability of Detection Vs. Range for B-II ----- 89

3.29 Searcher Position at Time of Detection for B-II - 90

3.30 Target Position at Time of Detection for B-II --- 91

3.31 Typical Figure-of-Merit for B-III --------------- 92

3.32 Cumulative Probability of Detection for B-III --- 93

3.33 Cumulative Probability of Counter-Detection for B-III -- 94

3.34 Probability of Detection Vs. Range for B-III ---- 95

3.35 Searcher Position at Time of Detection for B-III --- 96

3.36 Target Position at Time of Detection for B-III -- 97

3.37 Probability of Counter-Detection Vs. Range for B-III -- 98

3.38 Target Position at Time of Counter-Detection for B-III -- 99

3.39 Searcher Position at Time of Counter-Detection for B-III -- 100

3.40 Searcher Propagation Loss for Case C Examples --- 106

3.41 Target Propagation Loss for Case C Examples ----- 107
3.42 Searcher Track for Cases C-I and C-II ----------- 108
3.43 Cumulative Probability of Detection for C-I ----- 109
3.44 Target Position at Time of Detection for C-I ---- 110
3.45 Searcher Position at Time of Detection for C-I -- 111
3.46 Probability of Detection Vs. Range for C-I ------ 112
3.47 Cumulative Probability of Detection for C-II ---- 113
3.48 Searcher Position at Time of Detection for C-II - 114
3.49 Target Position at Time of Detection for C-II --- 115
3.50 Mean Time Ratio Vs. Speed Ratio ----------------- 116
3.51 Median Time Ratio Vs. Speed Ratio --------------- 117
F.1 The MSEL Model as a Markov Chain ---------------- 164

I. DESCRIPTION OF THE SIMULATION MODEL

A. INTRODUCTION

Although the body of analysis devoted to search theory is extensive (see, for instance, the lists of references in Koopman [Ref. 1] and Washburn [Ref. 2]), most of the techniques are applicable only to stationary targets. Recently there have been some important advances in the theory of search for moving targets (see Eagle [Ref. 3], Brown [Ref. 4], Stewart [Ref. 5], Stone [Ref. 6], and Washburn [Ref. 7]). Even so, it is still a very difficult, if not impossible, computational problem to calculate the probability of detection or the mean time to detection when the searcher has a speed close to that of the target and follows a realistic (i.e. not random) track. As a result, computer simulation of the search process is often used to evaluate the effectiveness of search tactics.

The Monte Carlo simulation computer program presented in this thesis is an attempt to model the passive acoustic search process in a one-searcher, one-target scenario where both the searcher and target platforms have sensors capable of detecting each other. The results of the simulation allows the analyst to closely approximate parameters (e.g. cumulative probability of detection, detection times, detection ranges) which are essential in planning search operations. With these results, the military planner can

make informed decisions concerning asset allocation and specific platform employment.

The computer program, Passive Acoustic Search Simulation (PASS), is based on an acoustic fluctuation model described by McCabe [Ref. 10:pp.7-23] and Hurley [Ref. 11:pp.25-34]. Precedessor programs include the FORTRAN program BEAR, and the ALGOL program SEARCH. BEAR was written by L. K. Arnold of D. H. Wagner, Associates (see Arnold [Ref. 8] for program documentation). SEARCH, a derivitive of BEAR, was written by W. J. Browning and J. Risberg, also of Wagner, Associates (see Browning [Ref. 9] for program documentation). SEARCH has been used extensively in operational and exercise environments under the direction of Submarine Development Squadron Twelve (SUBDEVRON-12), where it is currently resident on the Semi-Automatic Reconstruction Facility (SARF) computer. The basic structure of PASS is similar to that of BEAR, with some of the features of SEARCH added. Additionally, some modifications were made to algorithms in these earlier programs, and new features were added, including interactive data input and extensive use of graphical presentation of simulation results.

PASS is written in FORTRAN and is designed for use at the Naval Postgraduate School (NPGS). PASS is not portable in that it uses external subroutines in the Non-International Mathematics and Statistics Library (NONIMSL) and the DISSPLA graphics system. The subsequent sections of this chapter

are designed to give the reader an overview of the model without going into the specifics of the program code or logic. If the reader is interested only in learning how to use PASS, it is recommended that this chapter and Appendix A be studied. Appendix A is designed as a User's Guide and does not require any knowledge of the program code other than variable names, which are described in Appendix B. If the reader is interested in the structure of the program and the model, then this chapter and Chapter II should be studied. Some examples of varied applications of PASS are shown in Chapter III.

B. GENERAL PROGRAM DESCRIPTION

Program PASS is a Monte Carlo simulation of two passive sonar platforms in a search scenario. Arbitrarily, one platform is called the "searcher", and one the "target", where the target is typically thought of as a submarine, and the searcher as any passive sonar platform. Since both ships have passive sensors, the assignment of the names searcher and target does not connotate strict roles (i.e., the searcher may be detected first, and becomes a target in the truest sense!).

During each replication, the target's motion is partially random within a user-defined search area. The searcher motion is deterministic and controlled by user input. Each sensor is subject to random acoustic fluctuations

which induce random variations in detection ranges over time. Detection occurs whenever one platform obtains a positive signal excess, unless the user employs an integration time model, which requires weak signals to be present longer than strong signals for detection to occur.

The output of the program consists of distributions of parameters of interest (e.g. time of detection and range of detection), and point estimates derived from the statistics of these distributions (e.g. mean time to detection and mean detection range). The format of the output is numerical for point estimates, and graphical for distributions.

C. SEARCH SCENARIOS USED IN PASS

1. Area Search

The target is constrained to move within a specific rectangular area ("search area") in a "semi-random" fashion. Target speeds are distributed uniformly over a user specified speed range, and courses are uniform on (0, 360) degrees. Times between course and speed changes are exponentially distributed. The size of the search area and the parameters of the random motion are user specified. The searcher moves along a track which need not be contained within the search area. The searcher track and speeds are user specified.

2. Barrier Search

The target is constrained laterally to be within certain user specified bounds, thus simulating a "choke point".

Target motion through the choke point is generally vertical (i.e. from "top" to "bottom") with random course and speed deviations parameterized by the user. The searcher track is specified as in Area Search.

The distribution of the initial lateral position of the target can be specified by the user as either uniform across the choke point, or with specific probabilities as a function of distance across the choke point. In using the latter option, and by limiting the maximum course deviation, the user can control the distribution of the lateral target position when the target penetrates the barrier.

D. THE ENVIRONMENTAL MODEL

The environment is modeled in a conventional manner. PASS requires range/propagation loss information for both the searcher and target sensors. Direct path (DP) and convergence zone (CZ) data are entered separately. A cubic interpolation routine is used to determine propagation loss between the user-input data points. The convergence zones are modeled as inverted "square-wells" superimposed over the DP propagation loss curve. Up to five convergence zones are allowed.

E. THE DETECTION MODEL

In PASS a detection is always a "secure detection". That is, a detection by the searcher means that the searcher detected the target before the target detected the searcher.

Simultaneous detections are treated as special cases where neither the searcher or the target gets credit for a detection. The simulation terminates (replication ends) whenever:

1. The searcher makes a detection.
2. The target makes a detection (sometimes referred to as a counter-detection).
3. A simultaneous detection occurs.
4. In the barrier scenario, the target crosses a user specified lower boundary.
5. User specified maximum search time is exceeded.

The propagation loss curves are used in conjunction with searcher and target figure-of-merit (FOM) to determine detection ranges. The program uses FOM data which must be precalculated by the user based on specific platform source levels (Ls), backgroung noise (self noise and ambient noise) and sensor directivity (Le), and processor (machine and operator) recognition differential (Nrd).

1. Searcher Figure-of-Merit

The Ls of the target is assumed to be independent of target speed. In effect flow related noise and speed dependent propulsion noise are ignored. At the user's option, target Ls may be made a function of target aspect. It is assumed that the deterministic sensor performance of the searcher is affected by the searcher speed. This results in a searcher FOM that is dependent on searcher speed, and, optionally, target aspect.

2. Target Figure-of-Merit

The Ls of the searcher is assumed to be speed dependent, and independent of searcher aspect. This results in a target FOM that is dependent on searcher speed.

3. The Threshold Crossing Model With a Stochastic Error Process

Given a FOM against a target for a search platform, the maximum detection range of the searcher is determined from the propagation loss curves. This results in the sensor conforming to the "definite range law", or "cookie-cutter" performance. This type of sensor readily lends itself to mathematical analysis, but has the disadvantage of not reflecting actual sensor performance. What is seen under operational conditions is a variation in signal strength (and thus a variation in detection ranges) over time, even for targets at fixed range and source level.

To model these variations in received acoustic signal, a stochastic error process is added to the deterministic figure-of-merit. This results in the signal excess consisting of deterministic and random components. For a detailed discussion of threshold crossing models and associated stochastic error processes, see McCabe [Ref. 10] and Hurley [Ref. 11]. See Appendix D for a discussion of the relationship of signal excess (SE) to figure-of-merit (FOM).

The mean signal excess ($\overline{SE}$) is defined as the expected difference between the signal-to-noise ratio (SNR)

at the hydrophone output (in decibels, db) and the SNR that is estimated to afford a probability of detection of 0.5.

The model used for the "observed" signal excess, as a function of time in PASS, is of the form:

$$SE = \overline{SE} + X + Y \tag{1.1}$$

where SE is the deterministic mean signal excess, and X and Y are random variables that describe the fluctuations in the signal level.

If we let the subscript s apply to the searcher and t for target, the model used in PASS is:

$$SE_s(t) = \overline{SE}_s(t) + X(t) + Y(t) \tag{1.2}$$

$$SE_t(t) = \overline{SE}_t(t) + Y(t) + Z(t) \tag{1.3}$$

where:

1. X(t) is a stochastic error process which describes the fluctuations "local" to the searcher.
2. Z(t) is a stochastic error process which describes the fluctuations "local" to the target.
3. Y(t) is a stochastic error process which describes acoustic fluctuations of a global nature whose effect is common to the searcher and target.
4. X(t), Y(t), and Z(t) are independent processes.

X(t) and Z(t) may be thought of as modeling the fluctuations that affect only the performance of the detecting platform, such as onboard fluctuations in sensor/processor performance, including operator performance.

Y(t) may be thought of as modeling the gross acoustic fluctuations that occur along the "environmental path" between the platforms, such as temperature and salinity gradients and changes in bottom conditions.

With signal excess now a function of the mean signal excess and the stochastic effor processes, we have the means to model the behavior of the sensor by employing a threshold crossing detection model as follows:

1. If SE(t) is less than 0, the probability of detection is 0.0.
2. If SE(t) is greater than or equal to 0, the probability of detection is 1.0.

Thus, a detection occurs at time t if, and only if, SE(t) is greater than or equal to zero. The fact that SE(t) is now undergoing random fluctuations about the mean signal excess results in random variations in the sensor detection range.

F. STOCHASTIC ERROR PROCESSES

PASS allows the user to choose either a Lambda-Sigma Jump (LSJ) or a Gauss-Markov (GMA) error process to model the acoustic fluctuations. Hurley [Ref. 11] gives an overview of these processes, and others used in search simulation programs. McCabe [Ref. 10] provides a more mathematically complete treatment of the processes, along with a comprehensive comparison of the LSJ and GMA models. In PASS we use the sum of two LSJ or GMA process, frequently referred to as a compound process. In the following discussion, a simple process, vice the compound process is considered for clarity.

1. The Lambda-Sigma Jump Error Process

Define a Lambda-Sigma Jump error process, $\xi(t)$, as follows:

$$\underset{(s \le \tau)}{\xi(t+s)} = z(t)\xi(t) + [1 - z(t)]\eta \qquad (1.4a)$$

$$z(t) = \begin{cases} 1 \text{ if } s < \tau \\ 0 \text{ if } s = \tau \end{cases} \qquad (1.4b)$$

where:

1. τ is an exponential random variable with rate parameter λ.
2. η is a normal random variable with zero mean and variance σ^2

The process can be though of as a "marked" Poisson process with rate parameter λ, where the magnitude of the marks are themselves random variables with a normal distribution (with zero mean and variance σ^2). The standard deviation of the normal distribution, σ, can be thought of as a scale parameter in this process. The figure-of-merit, then, is constant over exponentially spaced time intervals, and changes to new levels determined by a normal distribution at the end of each of these time intervals.

The covariance function for $\xi(t)$, is:

$$\text{Cov}\,[\xi(t), \xi(t+s)] = \sigma^2 e^{-\lambda s} \qquad (1.5)$$

This function is a measure of the correlation between values of $\xi(t)$ at different times. From this we can see that $\xi(t)$ is second order stationary. That is, the covariance depends

on the time difference, s, and not the time, t. The parameter λ determines the amount of dependence between successive acoustic levels, and can be chosen to yield any result from independence to complete dependence. The resulting process is then a piecewise continuous function in which there are periods of complete dependence of detection opportunity interrupted by fluctuations that introduce independence between these periods.

In PASS, then, we must provide a total of six parameters to completely describe the compound error processes used to model the acoustic fluctuations. They are:

1. λ_1 = rate parameter for the searcher-local error process, X(t).
2. σ_1 = scale parameter for the searcher-local error process, X(t).
3. λ_2 = rate parameter for the global error process, Y(t).
4. σ_2 = scale parameter for the global error process, Y(t).
5. λ_3 = rate parameter for the target local error process, Z(t).
6. σ_3 = scale parameter for the target local error process, Z(t).

The selection of appropriate values for λ and σ is a not well understood function of the environment, sensor, and processor. In practice, the selection of the values are subjective decisions based on the experience of the analysist as what seems to yield reasonable results. Tehan [Ref. 12] provides a discussion of estimation techniques for

these parameters. In applying these recommendations to PASS, which has compound processes, the values of λ and σ were governed by:

$$\begin{array}{lcl} 3\le\lambda_1+\lambda_2\le5 & & 6\le(\sigma_1^2+\sigma_2^2)^{\frac{1}{2}}\le9 \\ & (1.6) & & (1.7) \\ 3\le\lambda_2+\lambda_3\le5 & & 6\le(\sigma_2^2+\sigma_3^2)^{\frac{1}{2}}\le9 \end{array}$$

See Appendix E for a description of the computer simulation of LSJ. McCabe [Ref. 10:pp.9-13] gives some analytical results based on the LSJ process.

2. The Gauss-Markov Error Process

The Gauss-Markov process, $\mu(t)$, introduces explicit dependence of the value of $\mu(t+s)$ based on the value of $\mu(t)$ for any s. The process is Markovian in that the present value depends only on the value that immediately proceeds it. The process is Gaussian in that, for any n, the joint distribution of $\{\xi(ti)\}$, i=1,...,n is multivariate-normal each with zero mean and variance σ^2. In simulating the process, we can use the functional form:

$$\mu(t+s) = e^{-\lambda s}\mu(t) + k\eta \qquad (1.8)$$

where:

1. $k = (1 - e^{-2\lambda s})^{\frac{1}{2}}$ (see Appendix E)
2. η is a normal random variable with zero mean and variance σ^2.

The process results in a continuous sample path, but is, in fact, nowhere differentiable. The desirability of using this type of error function in modeling acoustic fluctuations is that it is "more like reality". Whether it is in any sense "better" than the LSJ process has not been determined. Results of other simulations indicate that cumulative probabilities of detection are higher, and mean time to detect is lower, when the GMA process is used instead of the LSJ. However, the differences are not significant to the point of determining which process is most appropriate as an acoustic model. One drawback of the model is that it is not possible to exactly duplicate a continuous function on a digital computer, and thus the machine representation of the GMA is only a close approximation of the continuous error process. Thus, the manner in which the GMA process is digitally replicated has an effect on the results of the simulation.

The parameterization of the process is exactly analogous to that of the LSJ process. Again, $\lambda_{1,2,3}$ can be thought of as rate parameters, and $\sigma_{1,2,3}$ can be thought of as scale parameters, exactly as in the LSJ model. In some analyses, $1/\lambda$ is referred to as the "relaxation time" (and λ as the "relaxation coefficient") since it measures the decay time of statistical dependence between random variables in the fluctuation process for both the GMA and LSJ processes.

The covariance function for the Gauss-Markov process is exactly the same as that for the LSJ (see Equation 1.5), and therefore, the GMA is also a second order stationary process.

There are no analytical results similar to those from the LSJ process when the Gauss-Markov process is used. See Appendix E for a description of the computer representation of the GMA process.

G. A THREE-OUT-OF-FIVE DETECTION CRITERIA MODEL

McCabe [Ref. 10:p.50] contends that when a target is exposed for only a short amount of time but at a relatively high signal-to-noise ratio (SNR), the threshold crossing model returns an unrealistically high probability of detection, and presents a signal integration model based on a time dependent recognition differential. PASS has, as an option, a much simplified model which is easier, and faster to implement.

It seems necessary to alter the threshold crossing model under the above conditions so that detection would require some minimum combination of signal strength and time. This, in effect, would model what is actually integration time for the sonar processor. In general, the desired relationship is that the stronger a signal is, the shorter the time required for recognition as a valid contact. This model seems to be especially necessary when detection is based on other than aural cues, as is the case in modern sonars.

As a user option, instead of the threshold crossing model (where detection occurs the first time SE(t) is equal to or greater than zero), a 3-of-5 model can be employed which requires SE(t) $\geq$ 0 on 3 of the last 5 samples of signal level for detection. This Minimum Signal Excess Logic Model (MSEL) results in the desired effect of a weak signal being required to be "present" for a longer period for detection to occur. See Appendix F for a more complete analysis of the model.

McCabe [Ref. 10:p.50] recommends that an integration model be used whenever strong convergence zones are present.

H. A SEARCH MEASURE OF EFFECTIVENESS (MOE)

1. Searcher MOE

We may wish to measure the effectiveness of search (or evasion) in terms other than the probability of secure detection. For example, we may wish to penalize the searcher heavily for a secure target counter-detection, less heavily for a simultaneous detection, and still less heavily for no detection (escape).

Let:

1. Ns = the number of secure searcher detections
2. Nt = number of secure target detections
3. Nb = the number of simultaneous detections
4. Nn = number of no detection replications
5. Nr = the total number of simulation replications

Then define a MOE for the searcher as:

$$MOEs = \frac{Ns-3Nt-2Nb-Nm}{4Nr} + 0.75 \qquad (1.9)$$

where

$$0 \leq MOEs \leq 1 \text{ and } MOEs = 1 \Leftrightarrow Ns = Nr$$

Note that the weighting factors are a function of the relative penalties assigned to situations other than a secure detection, and could be easily adjusted.

2. Target MOE

In the area search scenario, we assume the target to be aggressive, and therefore a target MOE is assigned analogous to the searcher MOE:

$$MOE_t = \frac{Nt-3Ns-2Nb-Nm}{4Nr} + 0.75 \qquad (1.10)$$

In the barrier search, the target is assumed to want to avoid contact with the searcher, so the target MOE is adjusted as follows:

$$MOE_t = \frac{Nt-3Ns-2Nb+Nm}{4Nr} + 0.75 \qquad (1.11)$$

where, in both cases,

$$0 \leq MOE_t \leq 1 \text{ and } MOE_t = 1 \Leftrightarrow N_t = Nr$$

3. The Exchange Ratio

The exchange ratio is defined as:

$$ER = \frac{MOE_s}{MOE_t} \tag{1.12}$$

which provides a composite MOE for the search.

In area search:

$$ER>1 = Ns>Nt$$

$$ER = 1 \Leftrightarrow Ns=Nt$$

$$ER\ 1 \Leftrightarrow Ns<Nt$$

In barrier search:

$$ER>1 \Leftrightarrow Ns>Nt+\tfrac{1}{2}Nn$$

$$ER = 1 \Leftrightarrow Ns=Nt+\tfrac{1}{2}Nn$$

$$ER<1 \Leftrightarrow Ns<Nt+\tfrac{1}{2}Nn$$

I. PROGRAM RESULTS/OUTPUT AND STATISTICAL ANALYSIS

The following results are available from PASS:

1. A complete record of the input data.
2. Probability of detection by the searcher
    a. Fraction of detections which were direct path.
    b. Fraction of detections which were in the convergence zones.
    c. Fraction of detections in each convergence zone.
3. Probability of counter-detection by the target.
    a. Fraction of counter-detections which were DP.
    b. Fraction of counter-detections which were in the CZs.

c. Fraction of counter-detections in each CZ.

4. Histogram and statistics of time to detection by searcher.
5. Histogram and statistics of time to counter-detection by the target.
6. Histogram and statistics of range at detection by searcher.
7. Histogram and statistics of range at counter-detection by target.
8. Sectioning of time and range data (if sample size is large enough) to allow for estimates of the variability of sample statistics.
9. A plot of cumulative probability of detection versus time.
10. A plot of cumulative probability of counter-detection versus time.
11. A plot of conditional probability of detection versus range.
12. A plot of the searcher positions when the searcher detects the target.
13. A plot of the target positions when the searcher detects the target.
14. A plot of conditional probability of counter-detection versus range.
15. A plot of target positions when target counter-detects searcher.
16. A plot of the searcher positions when the target counter-detects the searcher.
17. A plot of typical searcher and target FOM for a five hour period.

J. VALIDATION OF THE PROGRAM

There is no existing body of data or analysis that would lend itself to a direct comparison to PASS results for a rigorous validity check. Numerous area and barrier search

scenarios, with various environmental, platform, and "random" parameters, were run with results not inconsistent with operational experience, limiting theoretical results, and other simulation program results. Due to the large number of variations the program allows, not all could be tested. The strongest statement that can be made regarding PASS validity is that it has not shown to be invalid after running in various configurations.

K. PROGRAM RUN TIME

PASS is a relatively expensive analysis tool from the standpoint of computer run time. On the NPGS computer (IBM VM-370 with three 3033 CPU's) a rule of thumb is that it will take about 50 CPU-seconds per 1000 replications of an area search in 100 nm x 100 nm search area. Obviously, more or less time will be required for different sensors and other scenarios. The point to be made is that the run time is long enough to warrant extensive pre-planning so that the maximum desired information is obtained from each run.

II. DESCRIPTION OF PROGRAM PASS

A. INTRODUCTION

The basic flow diagram for program PASS is shown in Figures 2.1, 2.2, and 2.3. Each "part" of the program, as described in the comment blocks in the source code (see Appendix H) is indicated in parentheses in each block of the flow diagram. The flow diagram, the annotated source code, and the description of the variable names (see Appendix B) should be used with this chapter to gain an understanding of the logic employed in the simulation and the implementation of this logic in the FORTRAN code.

B. DATA INPUT AND SIMULATION INITIALIZATION (PROGRAM PARTS 1, 2, 3)

All logical variables are set to default values prior to data input.

Data input is normally accomplished by the user responding to a number of options presented on the terminal screen. These options are displayed by sequential calls to subroutines OPTN0 through OPTN20. During this interactive session, the data necessary to run the program in the default mode is written to file 07 on the A-disk, which allows saving the basic data from one run to the next. The user can bypass all the options on the screen and proceed directly to program execution by reading the data from this

file through the subroutine READIT. Invoking subroutine READIT requires accepting the following defaults:

1. A complete, and properly formatted, input data set is assumed to be on the A-disk in file PASS DATA (file 07).
2. All input data will be sent to the output file PASS OUTPUT (file 06).
3. The acoustic fluctuation model is Lambda-Sigma Jump (LSJ).
4. A complete statistical analysis of the results is to be done.
5. The 3-of-5 detection criteria model is not used.
6. For area search the initial target position is distributed uniform on (0,XMAX), (0,YMAX).
7. For barrier search, the initial target lateral position is distributed uniform on (0,XMAX).
8. The searcher figure-of-merit is independent of target aspect.

Subroutines ECHO1 through ECHO6 write the input data to the output file if the user so desires.

The limits of the target course variations in the barrier scenario (ANG1, ANG2) are initialized if barrier search is selected. Detection counters (NDO, NDT, NCZDO(I), NCZDT(I), NBOTH, NONE) are set to zero, and the stack pointers (C1, C2, C3, C4) are initialized. The direction of each searcher path leg (DX(I), DY(I)) and the distance of each leg (DIST(I)) are computed, and the target speed range (STINC) and boundary reflection constants (TOXMAX, TOYMAX) are calculated.

C. REPLICATION COUNTER AND SETUP (PROGRAM PART 4)

If the required number of replications (NREP) have been completed, transfer is made to the data analysis and output routines (Part 18) and subsequent program termination. Otherwise, another replication is started.

A count of completed replications is written to the screen every 200 replications. Counters for the collection of a representative sample of figure-of-merit data (M1, M2) are reset if five hours of this data has not been collected in a previous replication. Simulation times (TNOW, TLAST) are set to zero, and signal MSEL counters (MSELO, MSELT) and convergence zone (CZ) detection pointers (KCZO, KCZT) are reset.

D. INITIALIZATION OF THE REPLICATION (PROGRAM PART 5)

At the start of each replication, the following initialization and setup is accomplished:

1. The time to the next fluctuation change (TIFL(I), I=1,2,3) is selected from an exponential distribution, parameterized by ALAM(I), by calls to subroutine EXPO.
2. The initial magnitude of the fluctuation (AFL(I)) is selected from a normal distribution, parameterized by SIGMA(I), by calls to the subroutine XLS.
3. The searcher position (XO, YO) is set to the first searcher track anchor point (XP(1), YP(1)).
4. The search leg pointer (NLEG) is set to one, and the search mode (MODE) is set to "drift". The time to the next searcher speed change (TSC) is set to the drift time. The figure-of-merit for both platforms (FOMO, FOMT) is selected based on searcher drift speed, and searcher speed (SO) is set to drift speed.

5. The time to the searcher course change (TCC) is set based on the first leg distance and drift speed, and searcher speed vectors (VX, VY) are set based on the direction vectors and search speed.

6. If the user is to specify the initial target lateral position in the barrier scenario, the initial target X-position (XT) is selected by a call to subroutine XDISTB. Otherwise, the initial target X-position is selected uniform on (0,XMAX) by a call to subroutine UZ1.

7. If the barrier scenario is selected, the initial target Y-position (YT) is selected a distance offset from YMAX. This offset distance is representative of the searcher being "time-late" on the barrier. The offset is calculated based on the maximum time late (START), the mean target speed, and a call to subroutine UZ1. In the area search scenario, the initial target Y-position is selected uniform on (0,YMAX) by a call to subroutine UZ1.

8. Target speed (ST) is selected uniform on (STMIN,STMAX) by a call to subroutine UZ1.

9. The time to target speed change (TTSC) is selected from the exponential distribution, parameterized by RTSC, by a call to subroutine EXPO.

10. In the barrier scenario, the initial target psuedo-course (X) is selected uniform on (ANG1,ANG2), and in the area search scenario, uniform on (0,360), by a call to subroutine UZ1. If the searcher FOM is dependent on target aspect, the target psuedo-course is saved (THETA). Target direction vectors (COSX, SINX) and speed vectors (UX, UY) are computed based on target course and speed.

11. The time to target course change (TTCC) is selected from an exponential distribution, parameterized by RTCC, by a call to subroutine EXPO.

E. DETERMINE DETECTION RANGES (PROGRAM PART 6)

If the searcher FOM is dependent on target aspect, the searcher FOM computed in Part 5 or 14 is re-calculated based on the current relative bearing from the target to the searcher (BREL). The subroutine RELB weturs BREL based on

the current searcher-target geometry. The subroutine INTRPL returns interpolated values of searcher FOM (FOMO) based on BREL and the searcher-speed/target-relative-bearing FOM data (FOMBD(I), FOMBS(I)). The searcher FOM (X) is computed by applying the error process values (AFL(1), AFL(2)) to FOMO. Five hours of searcher FOM (FO(I)) is saved for graphical display. The direct-path (DP) detection range for the searcher (RNGO) is obtained by applying the searcher FOM (X) to the DP propagation loss curve through subroutine INTRPL. If convergence zones are present (NCZO not zero), the searcher FOM (X) is applied to each CZ propagation loss value (CZLO(I)) to determine the most distant CZ in which a detection can take place (KCZO). If KCZO is zero, then no searcher detection can take place in a CZ.

The preceeding procedure, with the exception of the aspect dependency portion, is applied also to the target related data to produce values for RNGT and KCZT.

The maximum range at which a direct-path detection can take place by either platform (RMAX) is calculated from the maximum of RNGO, RNGT.

F. CHECK FOR DETECTION (PROGRAM PART 7)

The range between the platforms (RNG) is calculated based on the current geometry, and the algebraic difference between this range and the maximum DP detection range is calculated (TBIG). Direct-path detection is possible for at least one platform only if TBIG is less than or equal to zero.

If the searcher FOM will support detection in one or more CZs (KCZO not zero), the range (RNG) is checked to see if the target is in any searcher CZ annulus, starting from the most distant one that will support detection, and working in toward the searcher. If the target is in one of the CZs, KCZO is set to minus-one (as a CZ detection flag), and the CZ in which detection is possible (ICZO) is saved.

An identical CZ detection check for the target is done, and if the target can detect the searcher in a CZ, KCZT is set to minus one, and ICZT is saved.

The value of TBIG is set to the minimum distance the platforms would have to traverse (assuming a head-to-head closure) for a detection to just occur, based on the current searcher and target speeds and FOMs. That is, if the searcher and target were headed directly toward each other, TBIG would be the total distance covered by both platforms at the time of first detection by either.

If a CZ detection is possible (KCZO and/or KCZT less than zero) or if a DP detection is possible (TBIG less than zero), a transfer is made to Part 16A where it is determined if a detection is made, and by whom.

If detection is not possible, transfer is made to Part 8.

G. COMPUTE THE TIME OF THE "NEXT EVENT" (PROGRAM PART 8)

If the MSEL model is used, and either MSEL counter (MSELO, MSELT) is not in a zero state, this implies that detection

was possible at some earlier time, and now it is not. Therefore, any non-zero MSEL counter is decremented by one. If, after decrementing, any MSEL counter is not in a zero state, TBIG is set to zero. This results in the maximum simulation time-step defaulting to 0.05 hours.

The maximum simulation time-step is calculated by taking the maximum of 0.05 hours (3 minutes) or the time required for searcher and target to close the distance TBIG assuming all of the current platform speeds (SO, ST) are in the line of sight (head-to-head closure). The range TBIG is replaced by the time TBIG. The incremented time (TINC) is set to the current time (TNOW) plus TBIG, and the next event pointer (J) is set to 8. Next, the times of all other possible events are checked to see if any are earlier than TINC. If an event time is earlier than TINC, the time of this event replaces the current value of TINC, and the next event pointer is changed appropriately. When this process is completed, TINC will be the time of the "next event", and the value of J will identify what that event is. Note that if J remains at 8, the time increment will be TBIG. TBIG tends to be large when the platforms are far apart, and decreases as the range decreases.

If TINC exceeds the maximum allowable search time (TMAX), then transfer is made to Part 17 for the recording of no-detection data, and the start of a new replication, if appropriate.

H. MOVE THE SHIPS (PROGRAM PARTS 9A, 9B)

The time increment (X) is obtained by subtracting the current time from the incremented time, and the projected searcher position (XOT, YOT) is obtained by applying the time increment to the searcher velocity vectors (VX, VY). The distance from the current searcher position (XO, YO) to the projected searcher position (D1) is calculated from the geometry. The distance from the current searcher position to the next searcher track anchor point (D2) is calculated from the geometry. If D1 is less than or equal to D2, the searcher position is updated to the projected position. If D1 is greater than D2, then the time increment (X) is adjusted to put the searcher at the next anchor point, and the next event pointer (J) is set to 4 which will, in Part 12, cause the searcher position to be updated to the anchor point.

The target position (XT, YT) is updated by applying the time increment (X) to the target velocity vectors (UX, UY).

The simulation time (TNOW) is updated by the time increment (X).

If the target position, as calculated in Part 9A, ends up outside the search area (0,XMAX), (0,YMAX), the target position and velocity vectors are adjusted such that the boundaries appear to "reflect" the target. The target track behaves as would a light beam striking the mirrored surfaces of the search area boundaries. If the searcher FOM is target

aspect dependent, then the new target course must be saved (THETA) for use in subsequent calculation of relative bearing.

I. BRANCH TO THE "NEXT EVENT" (PROGRAM PART 10)

Depending on the value of the next event pointer (J), transfer is made to the program part which accomplishes the physical event associated with the time-step computed in Part 8.

If J=8 (which means "no event" takes place), transfer is made to Part 6 or Part 7, depending on the integration model in effect. If the MSEL (3-of-5) model is in effect, transfer must be made to Part 6 to reset the KCZO and KCZT flags. In future sections where transfer is made to Part 6 or 7, the same dependence on the MSEL detection model in effect will govern where transfer is made.

If J=1,2,3 then an acoustic fluctuation level is to change, and transfer is made to Part 11.

If J=4 then the searcher course is to change at a searcher track anchor point, and transfer is made to Part 12.

If J=5 then the target course is to change, and transfer is made to Part 13.

If J=6 then the target speed is to change, and transfer is made to Part 15.

If J=7 then the searcher speed is to change, and transfer is made to Part 14.

J. ACOUSTIC FLUCTUATION LEVEL CHANGE (PROGRAM PART 11)

The fluctuation level change is calculated differently depending on the model used. The next event pointer (J) also identifies which error term is to change.

In the LSJ model, a call to the subroutine XLS with input parameter SIGMA(J) returns a new fluctuation level (AFL(J)) from a normal distribution. Next, a call to the subroutine EXPO with input parameter ALAM(J) returns a new time increment (T) to the next fluctuation, which is added to current time to obtain the time ofthe next fluctuation (TIFI(J)). Transfer is then made to Part 6.

In the GMA model, the time to the next fluctuation is obtained in a manner identical to the LSJ model. The lag time from the last calculation of error signals (S) is obtained by subtracting the last time the levels were calculated (TLAST) from the current time, and then setting TLAST to the current time. All three of the signal fluctuations are calculated by successive calls to subroutine XLS parameterized by SIGMA(JJ), JJ=1,2,3, and the algebraic expression developed in Appendix E which involves ALAM(JJ), JJ=1,2,3. Transfer is then made to Part 6.

K. SEARCHER COURSE CHANGE (PROGRAM PART 12)

The searcher position (XO, YO) is assigned to the next search leg start point (XP(NLEG), YP(NLEG)) and new searcher velocity vectors (VX, VY) are calculated based on the next leg direction vectors (DX(NLEG), DY(NLEG)) and current

search speed (SO). The time to the next searcher course change (TCC) is calculated based on the leg distance (DIST(NLEG)) and current searcher speed. Transfer is made to Part 6 or 7, depending on the detection model in effect.

L. TARGET COURSE CHANGE (PROGRAM PART 13)

If the area search scenario is in effect, the new target course (X) is selected from a uniform distribution on (0,360) by a call to subroutine UZ1. If the barrier scenario is in effect, the new target course is selected from a uniform distribution on (ANG1,ANG2). The new target course is saved (THETA) if the searcher FOM is dependent on target aspect.

New target speed vectors (UX, UY) are calculated based on the new course and current target speed (ST). The time interval to the next target course change (T) is obtained from an exponential distribution by calling the subroutine EXPO with input parameter RTCC, and the time of the next course change (TTCC) is calculated based on this interval.

If the searcher FOM is a function of target aspect, transfer is made to Part 6 (via Part 11 to calculate new fluctuation levels if GMA is the fluctuation model) to determine new detection ranges. Otherwise, transfer is made to Part 6 or 7 depending on the detection model in effect.

M. SEARCHER SPEED CHANGE (PROGRAM PART 14)

The MODE state is changed to reflect the new searcher speed. The searcher and target FOM (FOMO, FOMT) are changed to reflect the new searcher speed, and the search speed (SO) is updated. The time of the next searcher speed change (TSC) is updated using the equivalence variable TIME(MODE). The time to the next searcher course change is updated based on the new search speed, and distance to the next searcher track anchor point (D). Transfer is made to Part 6 (via Part 11 if GMA is the fluctuation model) to determine new detection ranges.

N. TARGET SPEED CHANGE (PROGRAM PART 15)

The new target speed (ST) is selected from a uniform distribution on (STMIN, STMAX) by a call to the subroutine UZ1. The time interval to the next speed change (T) is selected from an exponential distribution by a call to the subroutine EXPO with input parameter RTSC, and the time of the next speed change (TTSC) is calculated based on this interval. New target speed vectors (UX, UY) are calculated based on the new speed and current target direction vectors (COSX, SINX). Transfer is then made to Part 6 or 7, depending on the detection model in effect.

O. DETERMINE WHICH PLATFORM CAN DETECT (PROGRAM PART 16A)

This part of the program is entered only if, in Part 7, detection by at least one of the platforms is possible.

Therefore, if the direct path detection range of the target (RNGT) is less than the current range (RNG), and the target cannot make a CZ detection on the searcher (KCZT greater or equal to zero), the detection opportunity must exclusively belong to the searcher, and transfer is made to Part 16C. Similarly, if the direct path detection range of the searcher (RNGO) is less than the current range, and the searcher cannot make a CZ detection on the target (KCZO greater or equal to zero), the detection opportunity must exclusively belong to the target, and transfer is made to Part 16D.

If neither of the above cases are true, then the detection must be simultaneous, and transfer is made to Part 16B.

P. SIMULTANEOUS DETECTION OPPORTUNITY (PROGRAM PART 16B)

If MSEL is not in effect, the simultaneous detection counter (NBOTH) is incremented, and transfer is made to Part 4 to start a new replication or output results.

If the MSEL model is used, both the searcher and target counters (MSELO, MSELT) are incremented. If both the counters equal three, then the actions in the preceeding paragraph are carried out. If only the searcher counter is at three, then transfer is made to Part 16C. If only the target counter is at three, transfer is made to Part 16D. If no detection takes place (both counters less than 3) transfer is made to Part 8 to determine the time-step to the next event.

Q. SEARCHER SECURE DETECTION OPPORTUNITY (PROGRAM PART 16C)

If the MSEL model is not used, the following takes place:

1. Searcher detection counter (NDO) is incremented.
2. If this was a CZ detection (KCZO = -1), the number of detections in the particular CZ is incremented (NCZDO(ICZO)).
3. The time of the detection is saved (NTDO(NDO)).
4. The range of the detection is saved (RNTDO(NDO)).
5. Searcher position at time of detection is saved (XODT(NDO), YODT(NDO)).
6. Target position at time of detection is saved (XTOD(NDO), YTOD(NDO)).
7. Transfer is then made to Part 4 to begin a new replication or output results.

If the MSEL model is used, the following takes place:

1. Searcher MSEL counter (MSELO) is incremented.
2. If MSELO is 3, the actions in the preceeding paragraph are executed.
3. If MSELO is less than three, the target MSEL counter (MSELT) is decremented if it was not zero, and transfer is made to Part 8.

R. TARGET SECURE DETECTION OPPORTUNITY (PROGRAM PART 16D)

If the MSEL model is not used, the following takes place:

1. Target detection counter (NDT) is incremented.
2. If this was a CZ detection (KCZT = -1), the number of detections in the particular CZ is incremented (NCZDT(ICZT)).
3. The time of the detection is saved (NTDT(NDT)).
4. The range of the detection is saved (RNTDT(NDT)).
5. Searcher position at time of detection is saved (XTDO(NDT), YTDO(NDT)).

6. Target position at time of detection is saved (XOTD(NDT), YOTD(NDT)).

7. Transfer is then made to Part 4 to begin a new replication or output results.

If the MSEL model is used, the following takes place:

1. Target MSEL counter (MSELT) is incremented.

2. If MSELT is 3, the actions in the preceeding paragraph are executed.

3. If MSELT is less than three, the searcher MSEL counter (MSELO) is decremented if it was not zero, and transfer is made to Part 8.

S. NO DETECTION OCCURS (PROGRAM PART 17)

If no detection occurs by the maximum allowed search time (TMAX), the no-detection counter (NONE) is incremented, and transfer is made to Part 4 to start a new replication or output results.

T. RESULTS OUTPUT (PROGRAM PART 18)

By calling subroutine SINKEM, the following output functions are accomplished:

1. Some results of the simulation (times, ranges) are written to file 06 on the A-disk in the form of histograms and statistics.

2. Some results of the simulation (times, ranges, cumulative probability of detection/counter-detection, geographical positions, and figures-of-merit) are written to file 10 on the A-disk. File 10 is designed to be a data file for the FORTRAN program PASPLT which produces graphical output by invoking the DISSPLA graphics system.

3. An abbreviated summary of the simulation results is sent to the terminal screen (file 08) and to file 06 on the A-disk.

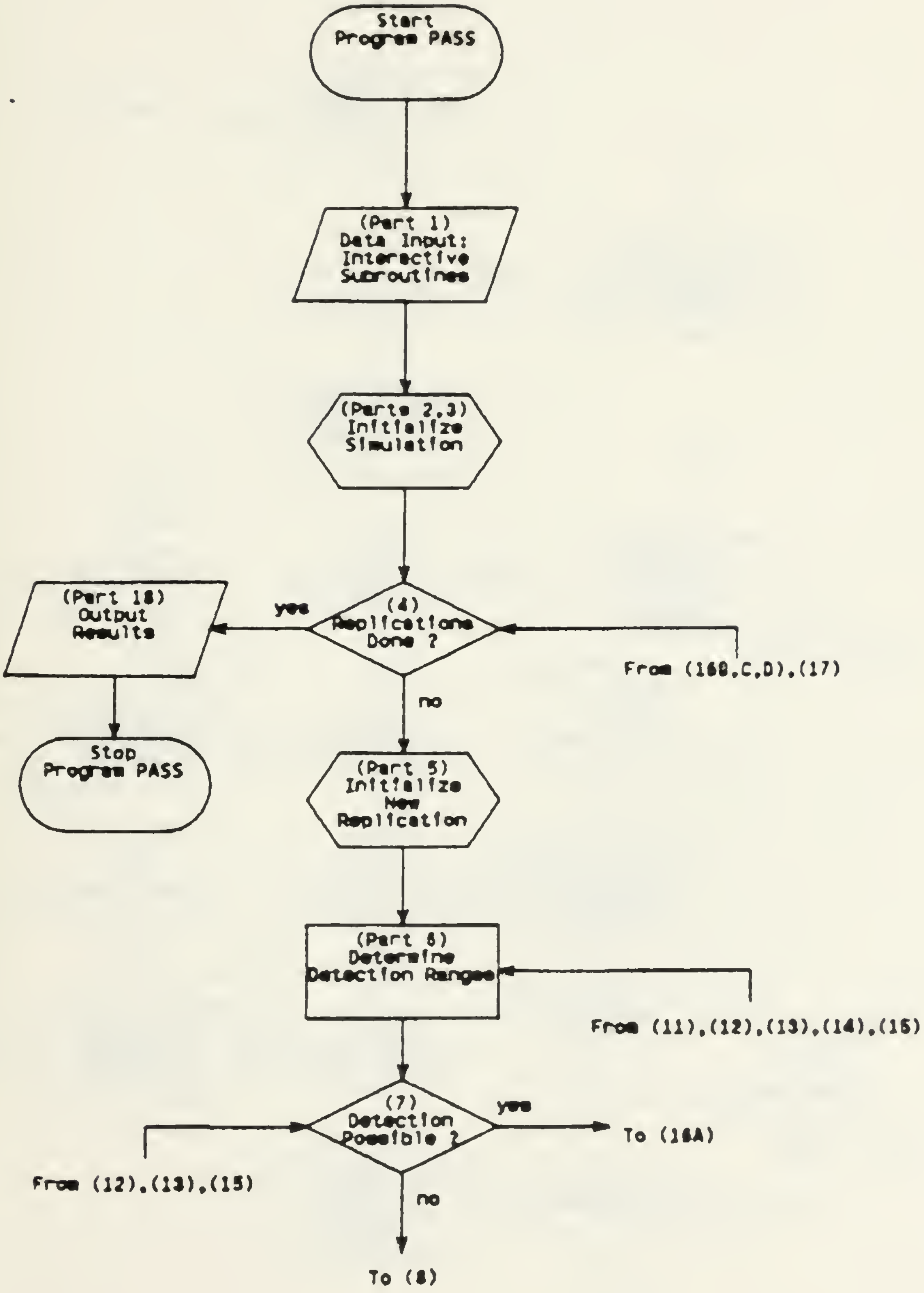

Figure 2.1. Flow Diagram for Program PASS Parts 1-7.

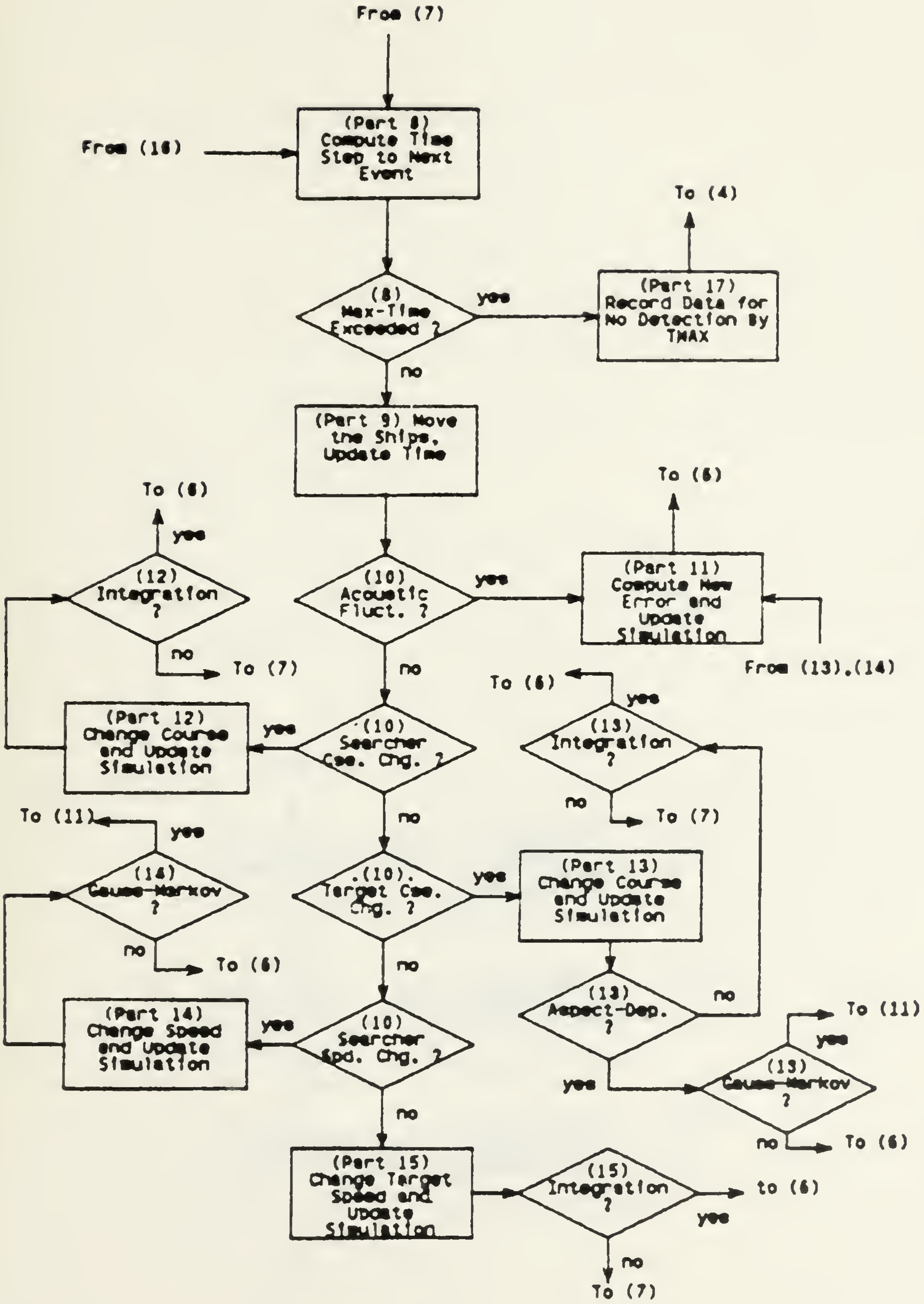
From (7)
(Part 8) Compute Time Step to Next Event
From (16)
To (4)
(8) Max-Time Exceeded ?
yes
(Part 17) Record Data for No Detection By TMAX
no
(Part 9) Move the Ships, Update Time
To (6)
To (6)
yes
(12) Integration ?
no
To (7)
(10) Acoustic Fluct. ?
yes
(Part 11) Compute New Error and Update Simulation
no
From (13),(14)
To (6)
yes
(Part 12) Change Course and Update Simulation
yes
(10) Searcher Cse. Chg. ?
(13) Integration ?
no
To (7)
no
To (11)
yes
(14) Gauss-Markov ?
no
To (6)
(10) Target Cse. Chg. ?
yes
(Part 13) Change Course and Update Simulation
no
(13) Aspect-Dep. ?
no
To (11)
(Part 14) Change Speed and Update Simulation
yes
(10) Searcher Spd. Chg. ?
yes
(13) Gauss-Markov ?
yes
no
no
To (6)
(Part 15) Change Target Speed and Update Simulation
(15) Integration ?
to (6)
yes
no
To (7)

Figure 2.2. Flow Diagram for Program PASS Parts 8-15.

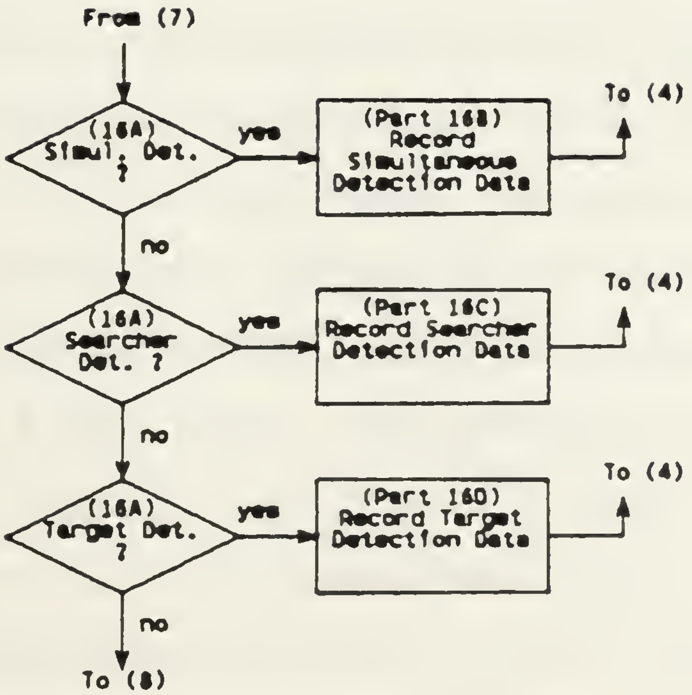
From (7)
(16A) Simul. Det. ?
yes
(Part 16B) Record Simultaneous Detection Data
To (4)
no
(16A) Searcher Det. ?
yes
(Part 16C) Record Searcher Detection Data
To (4)
no
(16A) Target Det. ?
yes
(Part 16D) Record Target Detection Data
To (4)
no
To (8)

Figure 2.3. Flow Diagram for Program Pass Part 16.

III. EXAMPLE SEARCH PROBLEMS

A. INTRODUCTION

In this chapter a set of search scenarios will be investigated to demonstrate the operation and capabilities of PASS. In each scenario (with the exception of the definite range law examples), environmental and performance parameters are representative of "real-world" conditions, but they do not represent any specific set of platforms or environmental conditions. This was done to preclude questions regarding security classification.

The purpose of this chapter is demonstration. As such, the problems investigated are treated in what may be called a superficial manner. That is, the questions posed are not rigorously analyzed, and conclusions drawn can only be applied within the narrow constructs of the specific scenario.

With one exception, the output from PASS (and PASPLT) as shown in the figures in this chapter is unaltered. The exception is the histogram and statistics output produced by the NONIMSL subroutine HISTGP, which is produced on the computer center line printer. The format of the histograms and statistical output is oversize, and is not of high enough quality to be photo-reduced for legible inclusion in this thesis. Selected statistical results from HISTGP will be provided in tabular form when required.

B. AREA SEARCH: NOISE QUIETING VS. IMPROVED SENSOR PERFORMANCE

1. Scenario Description

A target with a relatively large acoustic disadvantage is the subject of systematic area search. The question examined is which of the following possibilities would be more advantageous to the target:

1. Decreasing target radiated noise by 3 db, or
2. Increasing the FOM of the target sonar by 3 db.

2. PASS Input Data

Table I shows the input data for the base case (case A-I), in the format presented in the PASS output file.

3. Environment and Search Plan

Figures 3.1 and 3.2 show the propagation loss models used in the example. Note that even before figure-of-merit is calculated, the searcher has an advantageous situation in that his propagation loss curve predicts longer detection ranges than for the target (for the same FOM). This occurs because the geometric mean frequency of the searcher sonar is less than that of the target. Due to the presense of convergence zones, the MSEL model is used.

Figure 3.3 shows the searcher track, with track spacing of 75 miles (about equal to the range of the second CZ). The search area is 300 nm square.

4. Platform Parameters

The search speed is established as constant at 10 knots by setting sprint and drift speeds equal to 10 knots,

and searcher sprint and drift FOM equal at 87 db. Whenever constant speed search is to be simulated, the times at sprint and drift speeds should be set to some large number (500 hours in this case) to preclude unnecessary program steps.

The target speed is uniformly distributed between 5 and 10 knots, with changes in course and speed taking place on the average of twice per hour. The target FOM is 84 db in the base case.

5. Simulation Run Parameters

Two thousand replications were run, and a maximum search time was set at 720 hours (30 days). The random number seed can be any integer, and for these runs was selected as the time of program execution.

6. Results and Conclusions

Numerical results of the base case (Case A-I) and modified scenarios (Cases A-II, A-III, and A-IV) are shown in Table II. The modified scenarios reflect the following changes to the basic case:

1. Case A-II: The target platform gets a 3 db gain in sensor performance, which results in the target and searcher having equal FOM at 87 db.
2. Case A-III: The target platform reduces its radiated noise by 3 db which results in the target and searcher having equal FOM at 84 db.
3. Case A-IV: For comparison purposes, the target is assumed to achieve both the improvements in the above cases, resulting in a target FOM of 87 db, and a searcher FOM of 84 db.

The simulation results indicate that Case A-III is slightly more favorable to the target than Case A-II in the following respects:

1. The cumulative probability of (secure) detection by the searcher is smaller.
2. A larger proportion of the searcher detections are direct path, resulting in decreased mean detection range, and increased probability of counter-detection.
3. Less detections take place in the most distant CZ.
4. The change in the exchange ratio in A-III (92.2) is greater than that for A-II (83.2).
5. The mean time to detection is longer.
6. The mean and median detection ranges are shorter.

The conclusion is that a greater expected increase in performance will be realized if the 3 db gain is obtained by reducing radiated noise. Because of the better performance of the searcher sonar, the average slope of the propagation loss curve (change in prop. loss/change in detection range) will be less than that for the target (i.e. a "flatter" curve). Thus, the change in detection range per 3 db change would be greater for the searcher than the target. Therefore, reducing the searcher effectiveness (by quieting the target) has a larger effect on detection performance than increasing the target effectiveness (by improving the sensor) has on counter-detection performance.

For comparison purposes, a complete set of graphical output from PASS for Case A-I is shown in Figure 3.4 through 3.8, and for Case A-III in Figures 3.9 through 3.17.

TABLE I

INPUT DATA FOR CASE A-I

Search Area Dimensions:

XMAX = 300.00
YMAX = 300.00

Searcher Track Anchor Points:

NP = 7 KP = 2
XP(1) = 225.00 YP(1) = 0.0
XP(2) = 225.00 YP(2) = 75.00
XP(3) = 225.00 YP(3) = 225.00
XP(4) = 150.00 YP(4) = 225.00
XP(5) = 150.00 YP(5) = 75.00
XP(6) = 75.00 YP(6) = 75.00
XP(7) = 75.00 YP(7) = 225.00

Searcher Propagation Loss:

RO(1) = 2.50 OL(1) = 70.00
RO(2) = 5.00 OL(2) = 85.00
RO(3) = 7.50 OL(3) = 92.90
RO(4) = 10.00 OL(4) = 93.10
RO(5) = 20.00 OL(5) = 94.00
RO(6) = 30.00 OL(6) = 96.00
RO(7) = 45.00 OL(7) = 104.00
RO(8) = 60.00 OL(8) = 108.00
RO(9) = 85.00 OL(9) = 116.00
RO(10) = 95.00 OL(10) = 120.00

Target Propagation Loss:

RT(1) = 2.50 TL(1) = 70.00
RT(2) = 5.00 TL(2) = 95.00
RT(3) = 7.50 TL(3) = 104.00
RT(4) = 10.00 TL(4) = 105.00
RT(5) = 20.00 TL(5) = 109.00
RT(6) = 30.00 TL(6) = 115.00
RT(7) = 45.00 TL(7) = 130.00
RT(8) = 60.00 TL(8) = 136.00
RT(9) = 85.00 TL(9) = 146.00
RT(10) = 95.00 TL(10) = 150.00

Searcher Convergence Zones:

RCZO(11) = 32.00 RCZO(12) = 42.00 CZLO(1) = 88.00
RCZO(21) = 68.00 RCZO(22) = 80.00 CZLO(2) = 90.00

Target Convergence Zones:

RCZT(11) = 33.00 RCZT(12) = 42.00 CZLT(1) = 92.00
RCZT(21) = 69.00 RCZT(22) = 79.00 CZLT(2) = 103.00

Remaining Platform and Run Parameters:

FOMOD = 87.00 FOMOS = 87.00 SOD = 10.00 SOS = 10.00 TD = 500.00 TS = 500.00 FOMTD = 84.00
FOMTS = 84.00 STMIN = 5.00 STMAX = 10.00 RTSC = 2.00 RTCC = 2.00 SEED = 84439 NREP = 2000
TMAX = 720.00 LAMBDA(1) = 3.00 LAMBDA(2) = 2.00 LAMBDA(3) = 3.00 SIGMA(1) = 3.00 SIGMA(2) = 6.00
SIGMA(3) = 3.00 Compound Error Function Correlation = 0.8000

TABLE II

NUMERICAL RESULTS FOR CASE A EXAMPLES

	Case A-I	Case A-II	Case A-III	Case A-IV
PD	0.9725	0.9375	0.9210	0.8490
PDDP	0.1496	0.1515	0.1900	0.1914
PDCZ	0.8504	0.8485	0.8100	0.8086
PDCZ1	0.1172	0.0960	0.1482	0.0960
PDCZ2	0.7322	0.7525	0.6618	0.7126
MOES	0.9790	0.9502	0.9370	0.8732
PCD	0.0015	0.0115	0.0150	0.0540
PCDDP	0.3333	0.0000	0.0000	0.0093
PCDCZ	0.6667	1.0000	1.0000	0.9907
PCDCZ1	0.6667	1.0000	1.0000	0.9907
PCDCZ2	0.0000	0.0000	0.0000	0.0000
MOET	0.0080	0.0243	0.0310	0.0783
ER	122.4	39.19	30.20	11.16
(T)s	33.7	35.2	41.8	42.9
(T)t	26.4	17.8	32.3	33.9
(R)s	68.9	69.7	64.6	67.7
(R-50)s	75.6	75.6	73.1	74.1
(R)t	33.0	38.0	37.9	38.3
(R-50)t	4.75	39.1	37.9	39.2

Notes:

1. Due to the low number of counter-detections in Cases A-I and A-II (3 and 23 respectively) the variance of the target parameter estimates will be large. Therefore, comparison should be based on searcher parameter estimates.
2. See Appendix C for explanation of abbreviations.

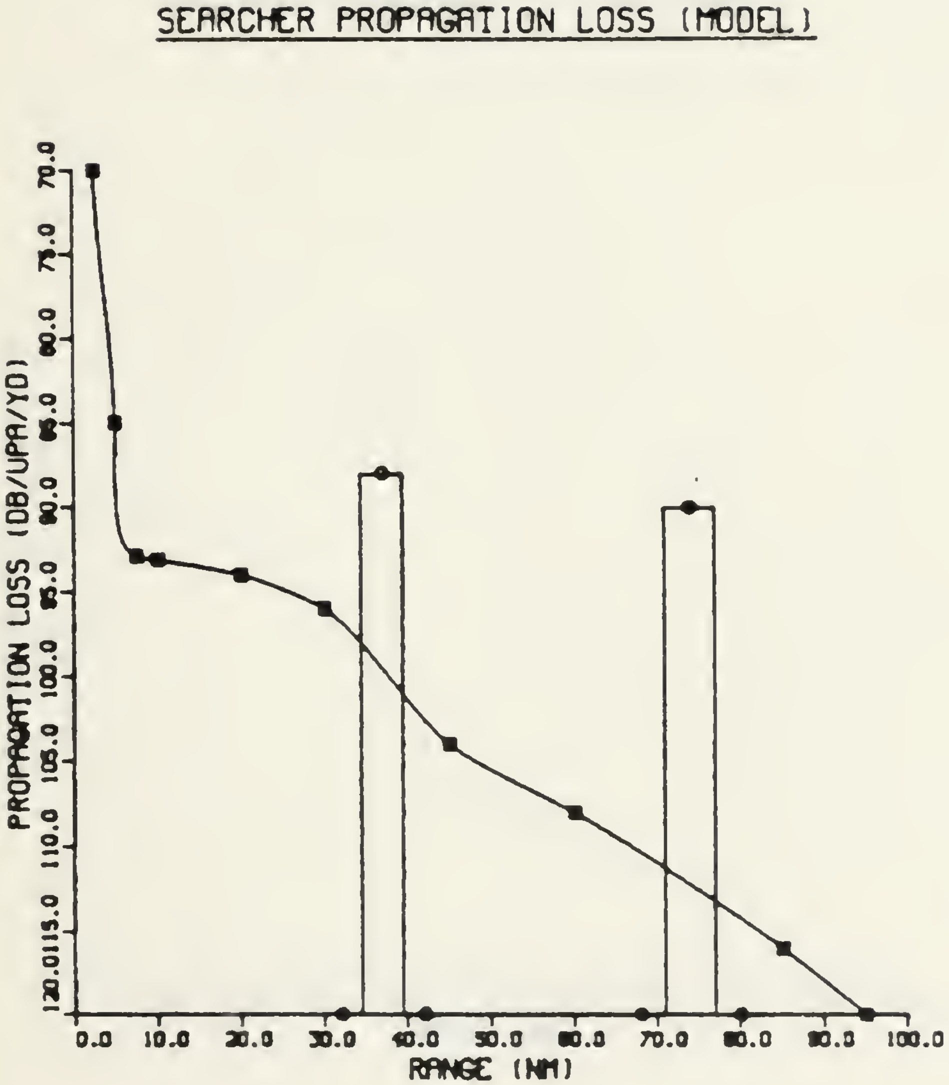

Figure 3.1. Searcher Propagation Loss Curve for Case A Examples.

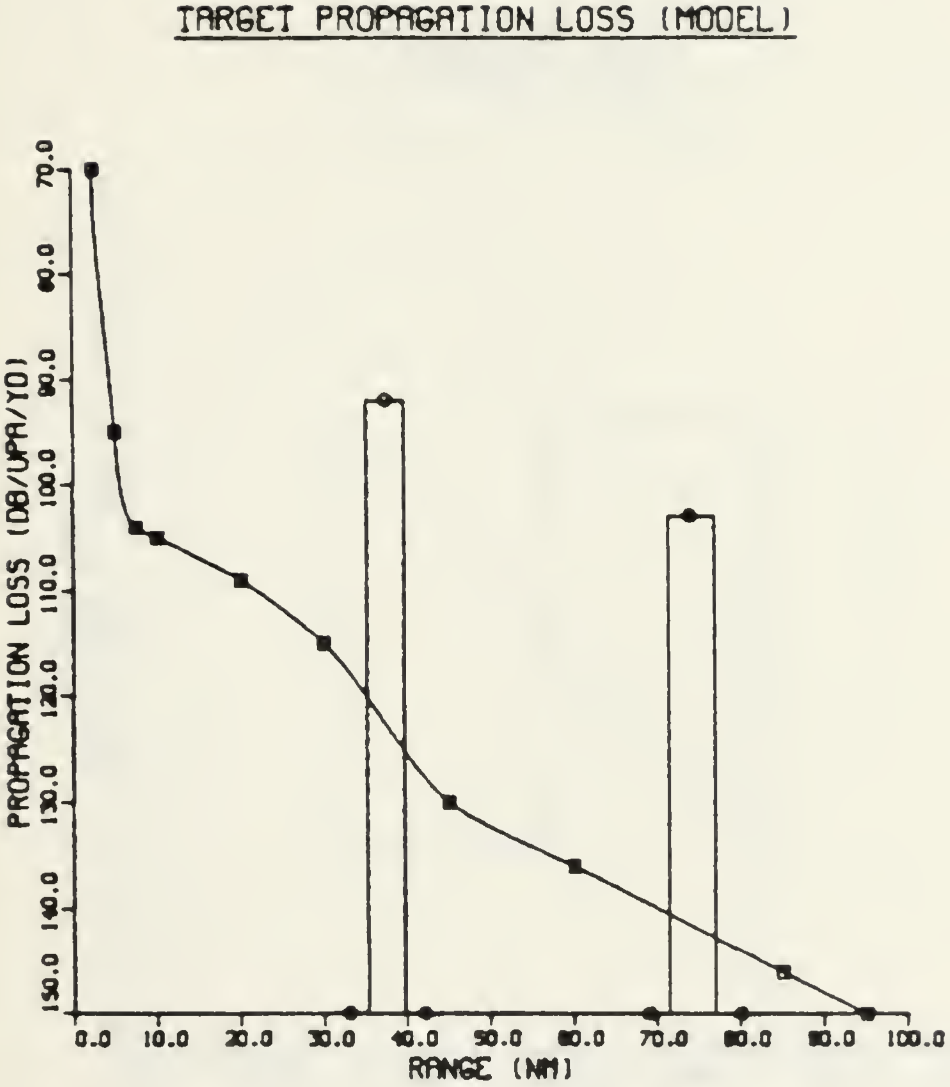

Figure 3.2. Target Propagation Loss Curve for Case A Examples.

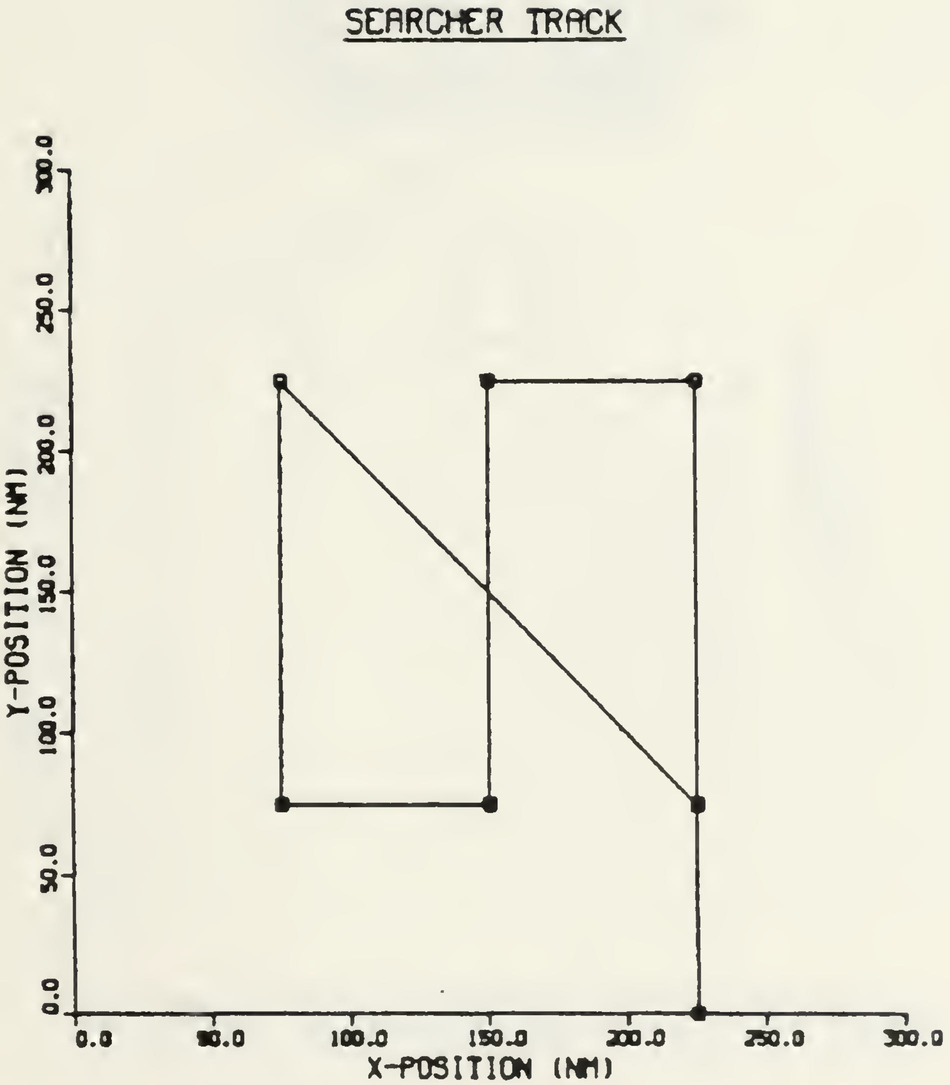

Figure 3.3. Searcher Track for Case A Examples.

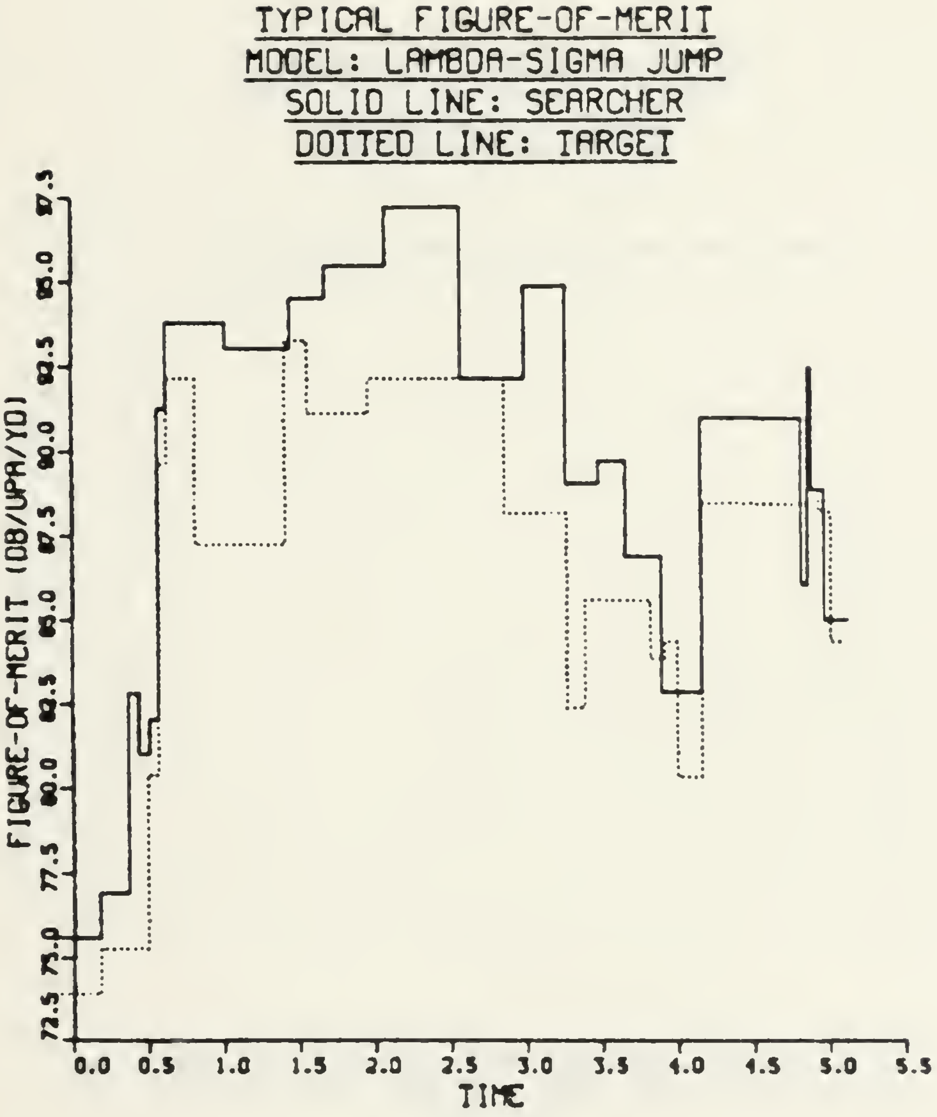

Figure 3.4. Typical Figures-of-Merit for A-I.

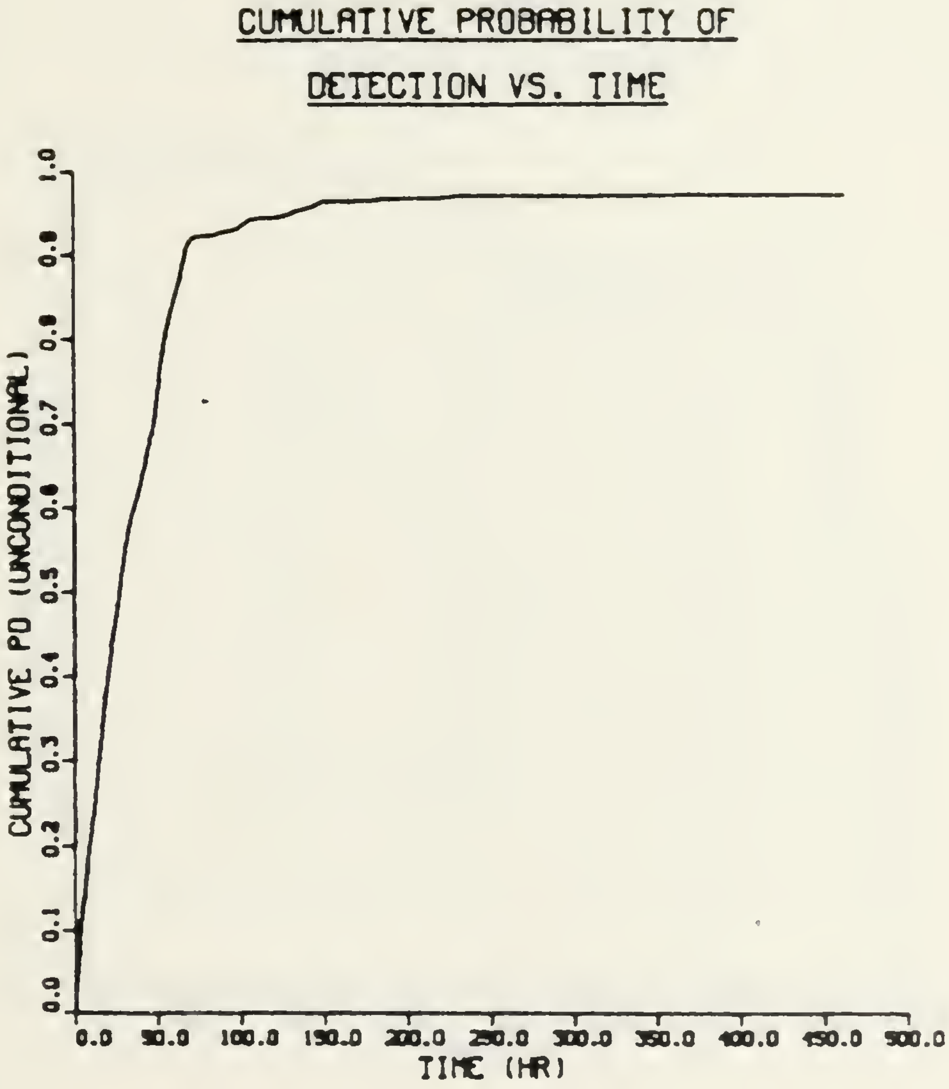

Figure 3.5. Cumulative Probability of Detection for A-I.

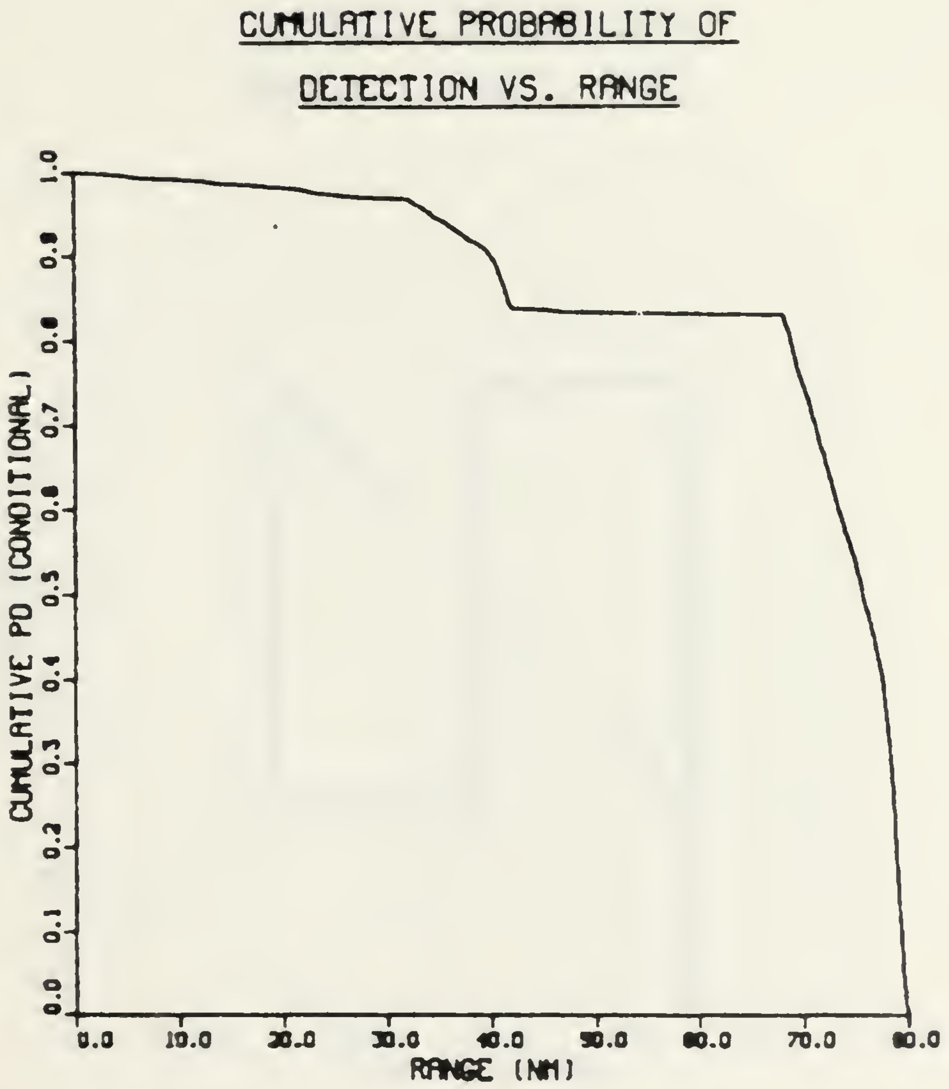

Figure 3.6. Progability of Detection Vs. Range for A-I.

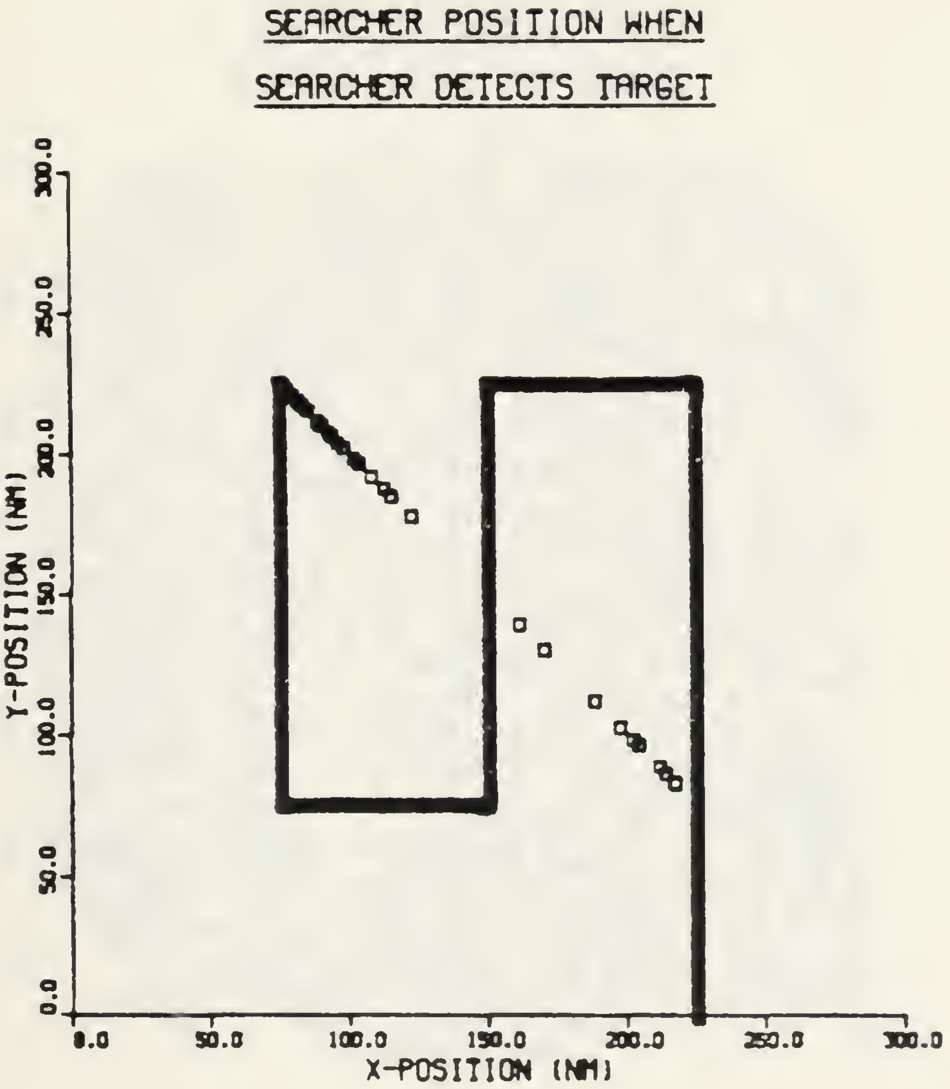

Figure 3.7. Searcher Position at time of detection for A-I.

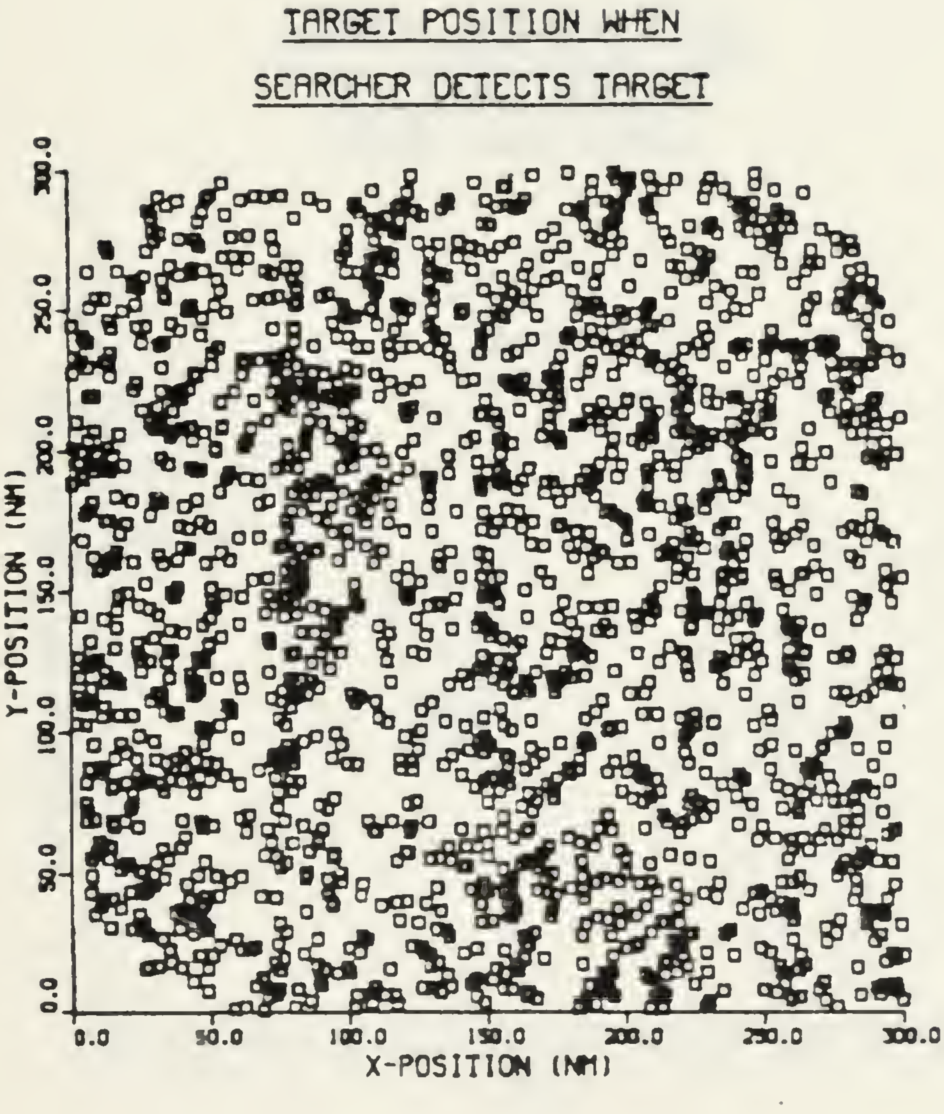

Figure 3.8. Target Position at Time of Detection for A-I.

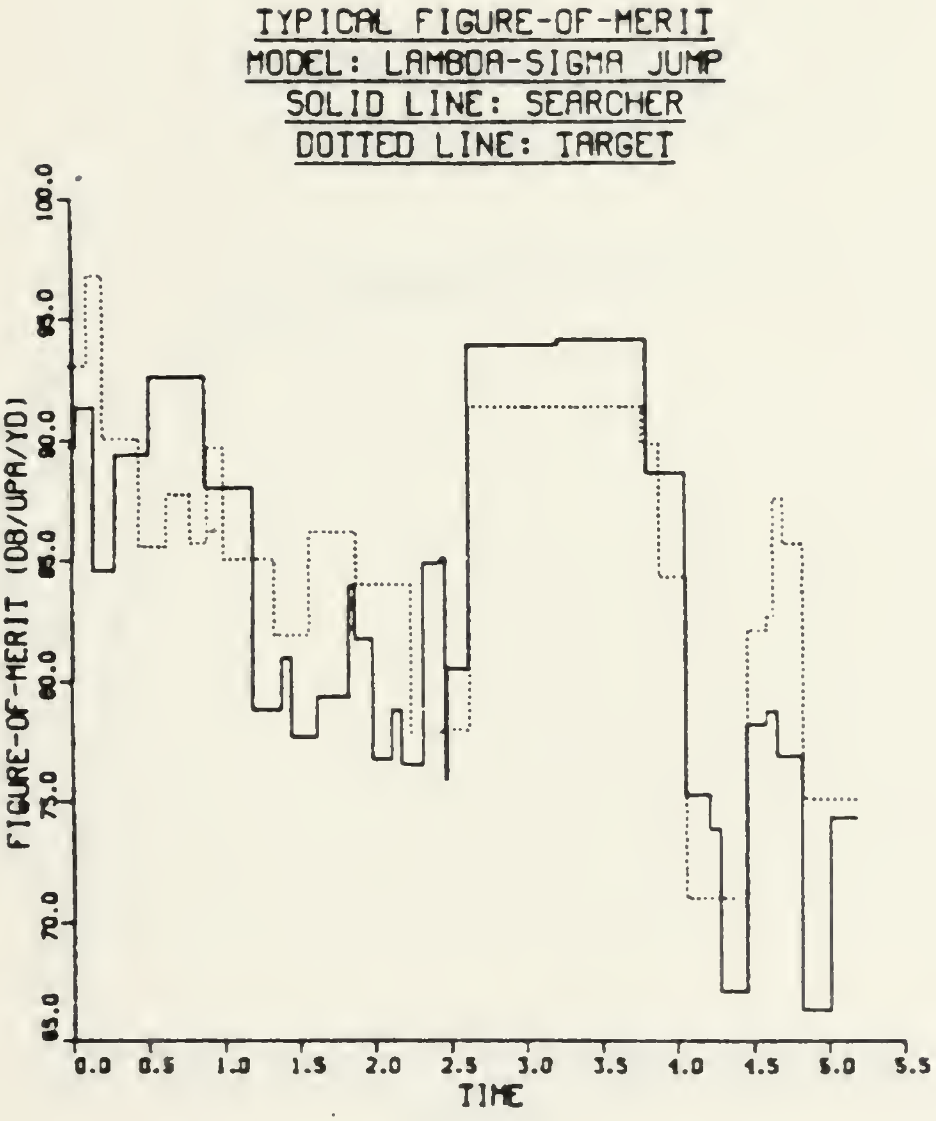

Figure 3.9. Typical Figure-of-Merit for A-III.

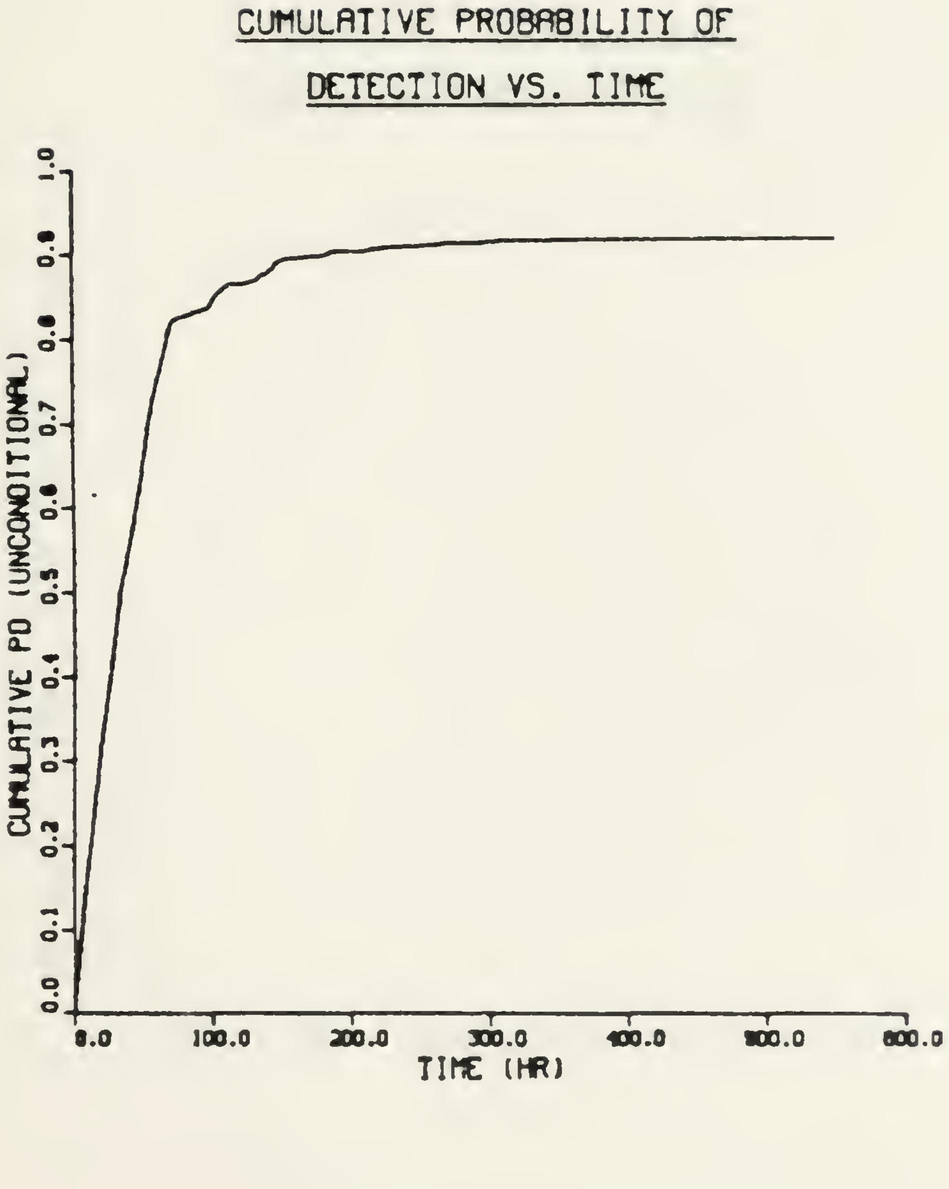

Figure 3.10. Cumulative Probability of Detection for A-III.

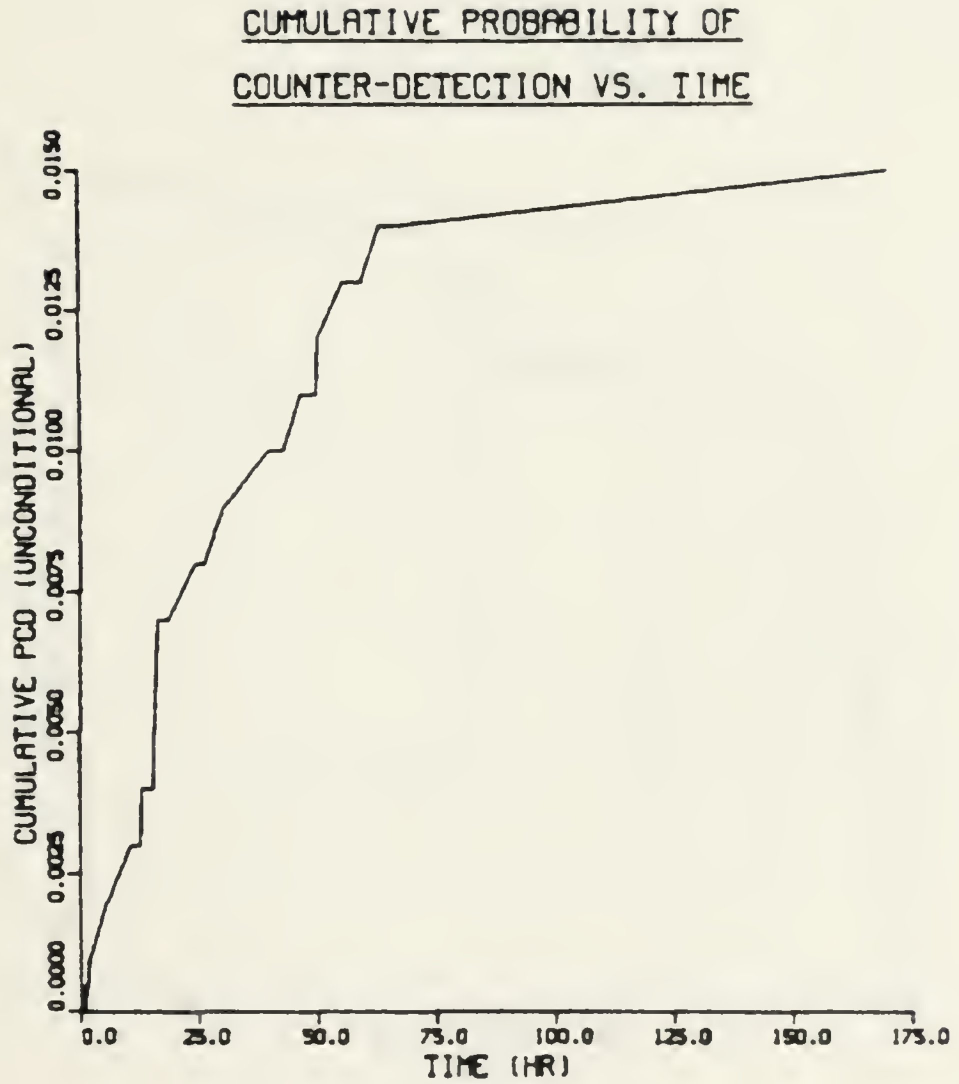

Figure 3.11. Cumulative Probability of Counter-Detection A-III.

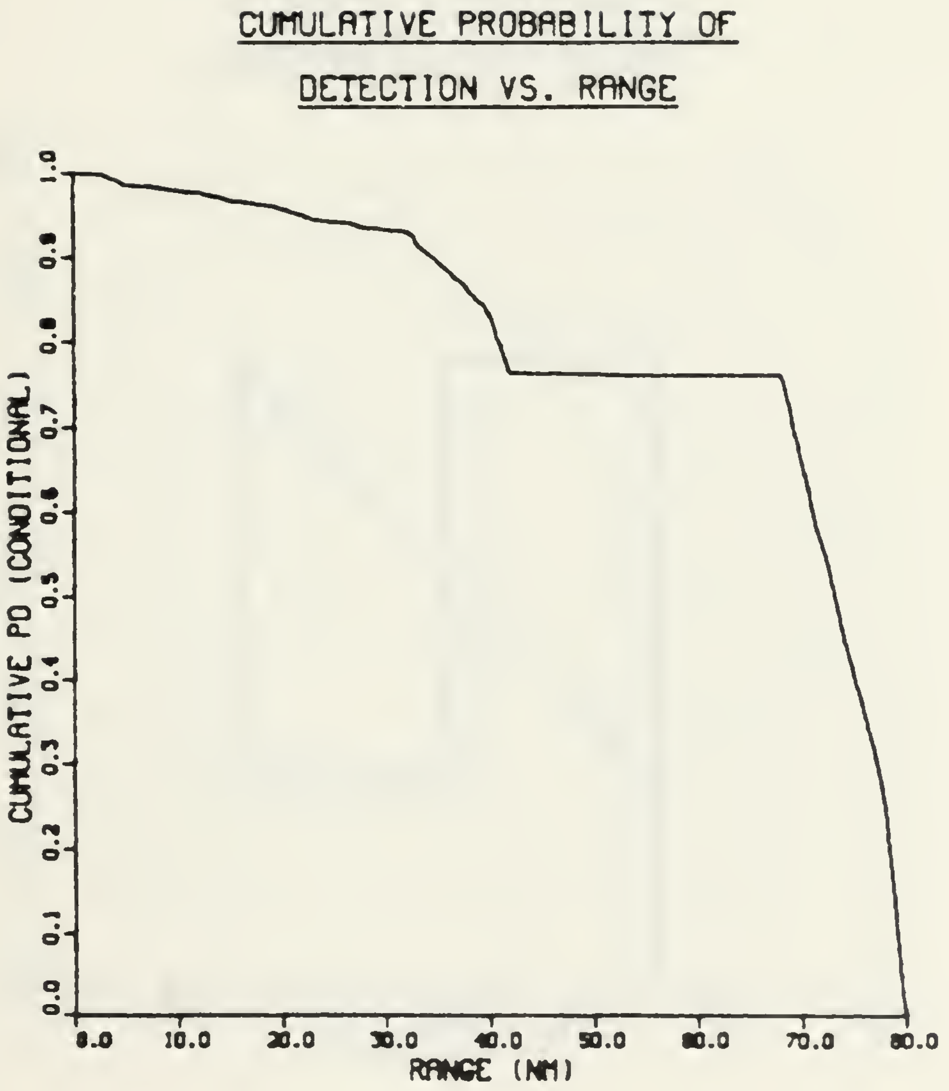

Figure 3.12. Probability of detection Vs. Range for A-III.

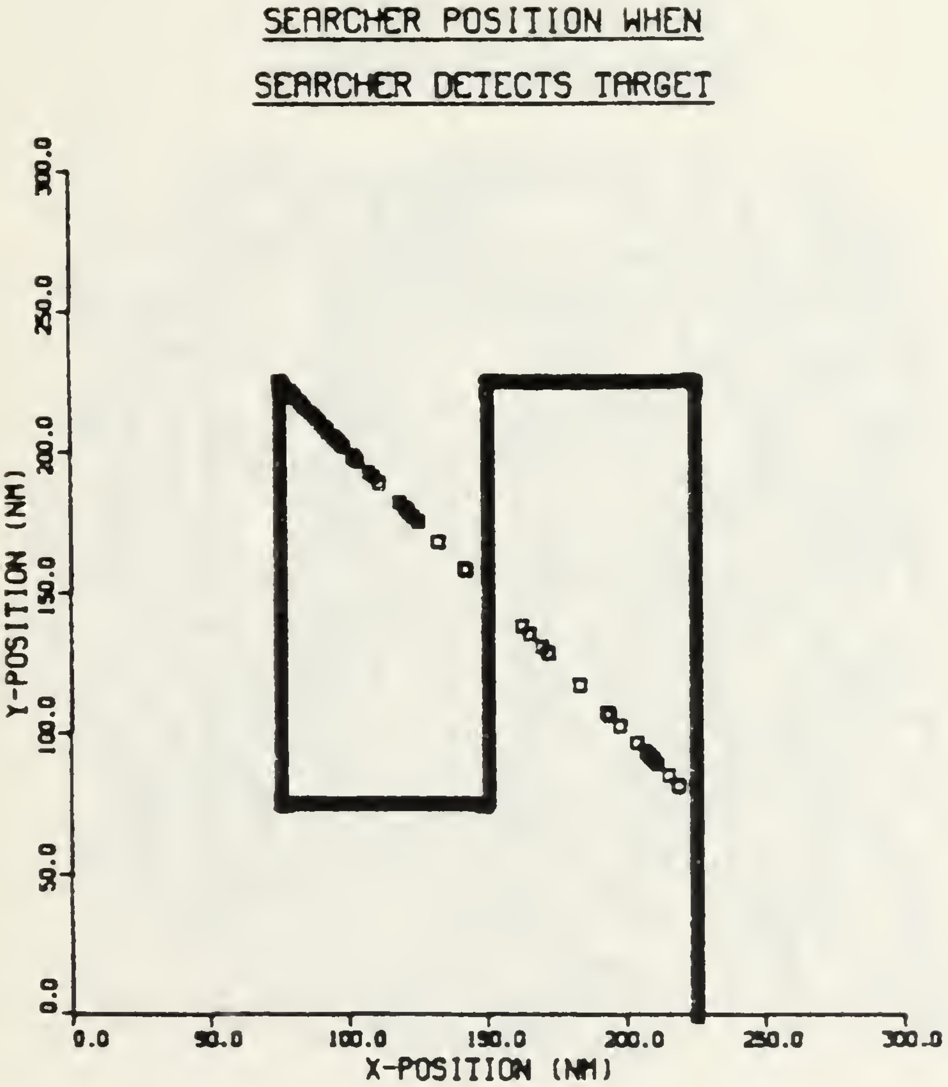

Figure 3.13. Searcher Position at Time of Detection for A-III.

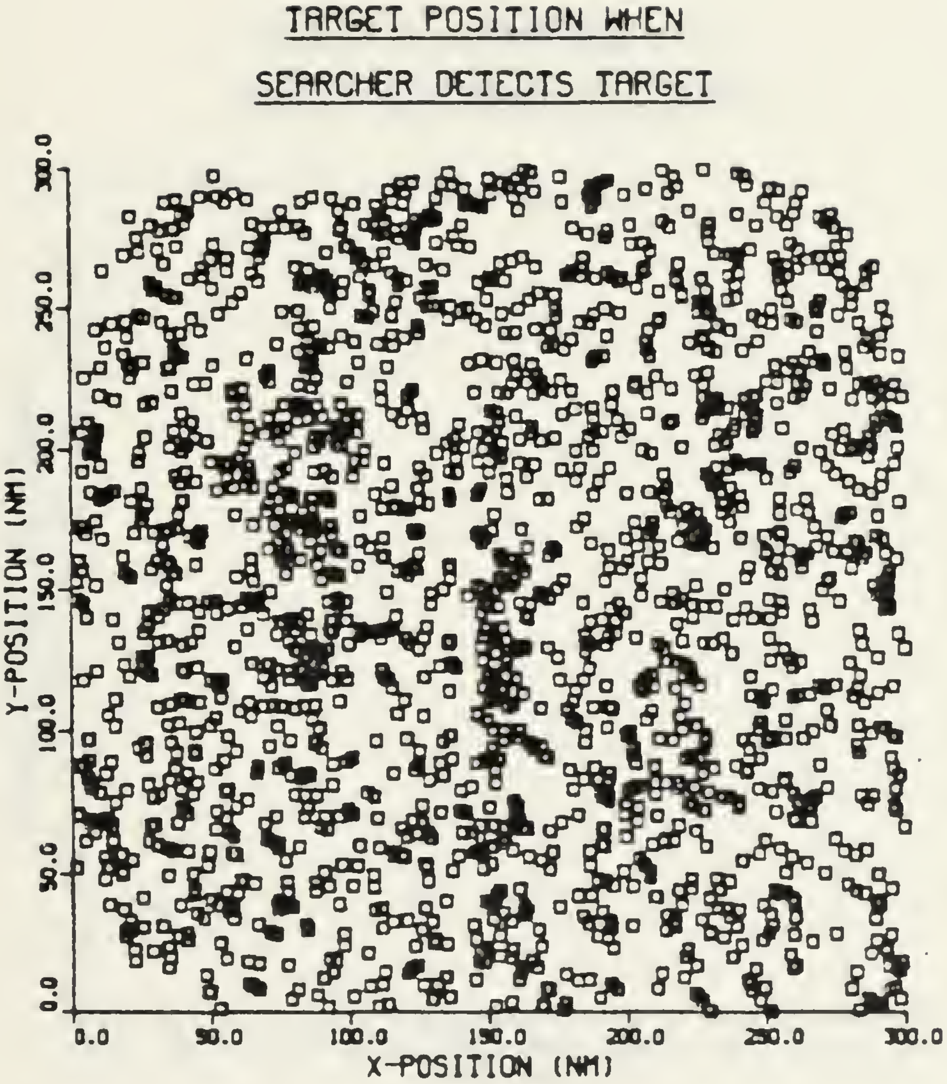

Figure 3.14. Target Position at Time of Detection for A-III.

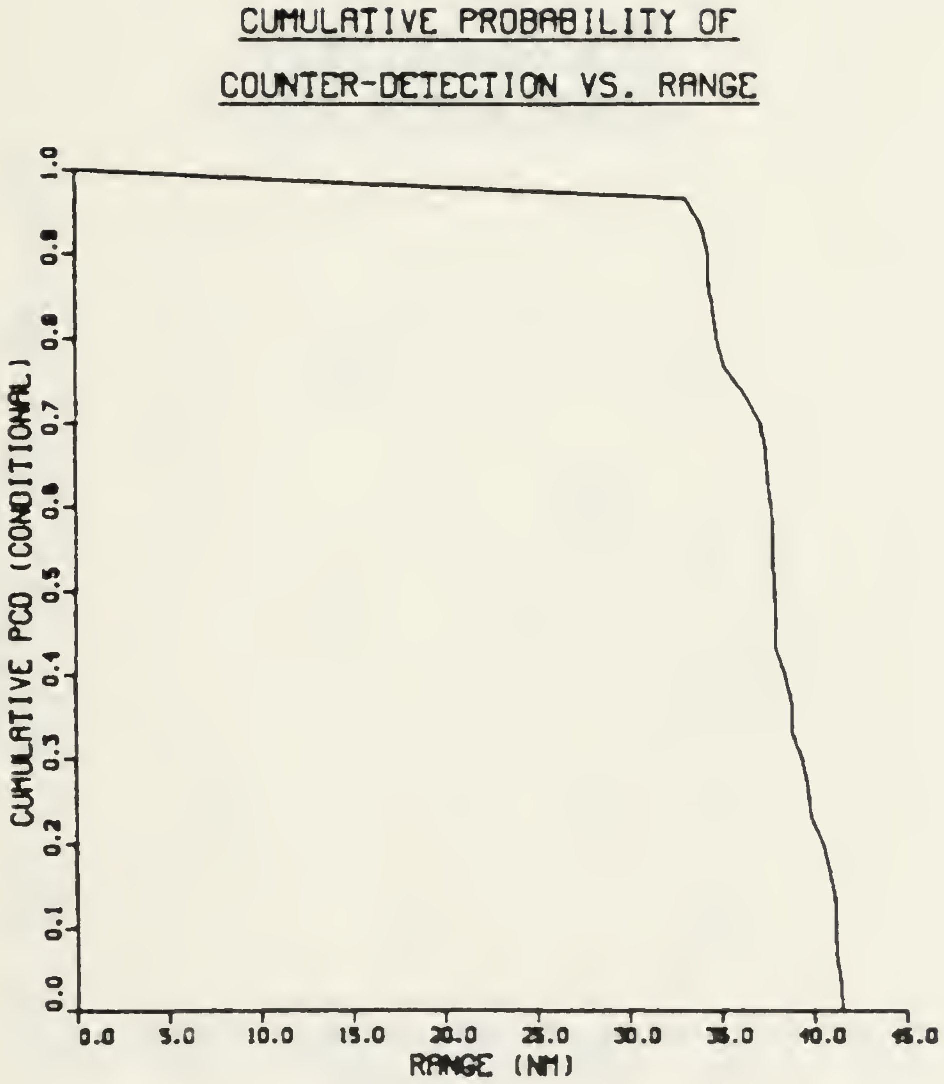

Figure 3.15. Probability of Counter-Detection Vs. Range for A-III.

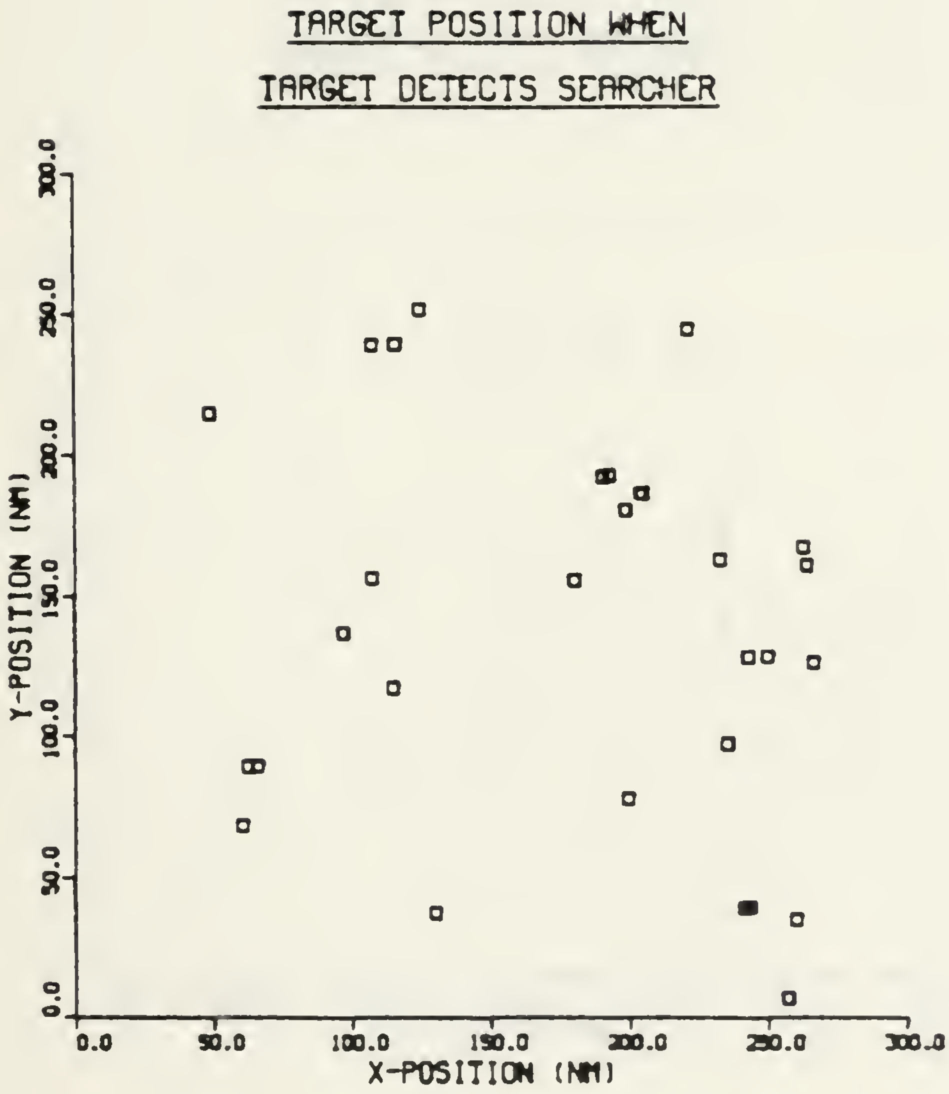

Figure 3.16. Target Position at Time of Counter-Detection for A-III.

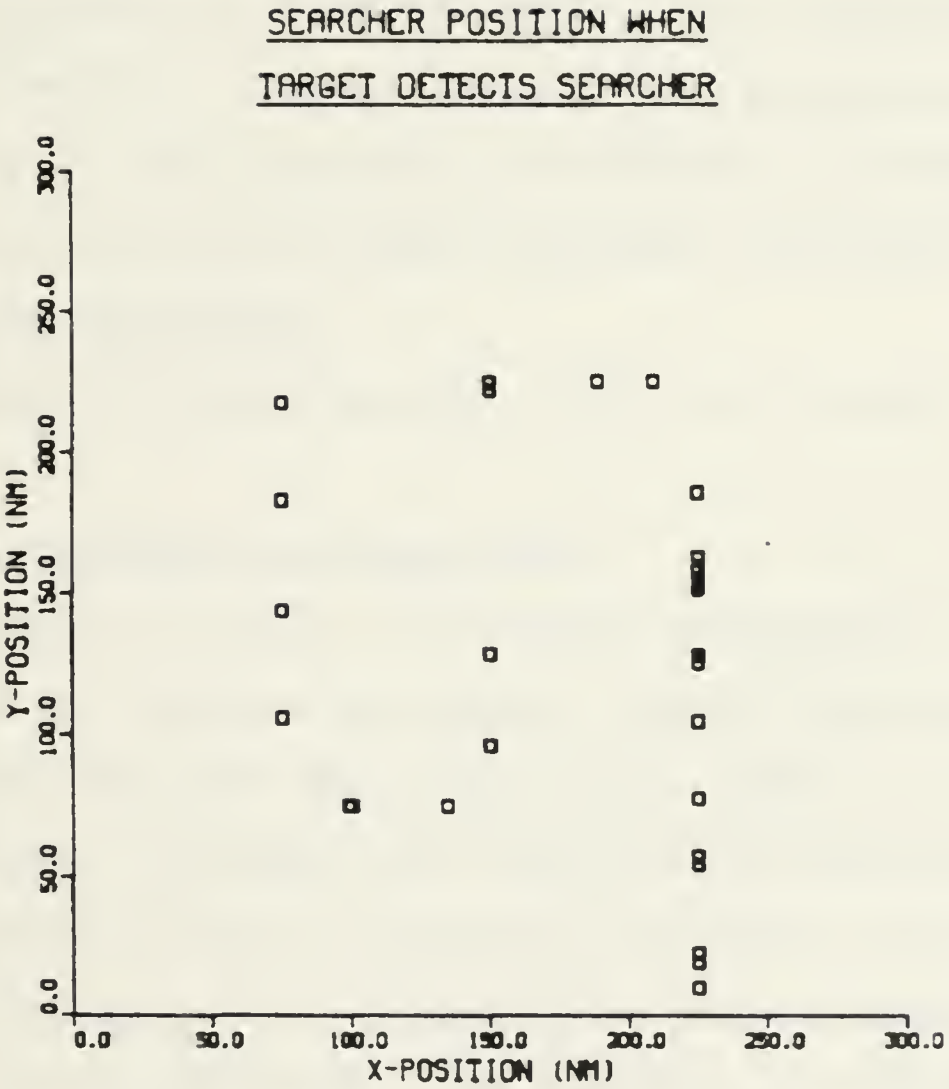

Figure 3.17. Searcher Position at Time of Counter-Detection for A-III.

C. BARRIER SEARCH: SOME FACTORS AFFECTING PENETRATION SUCCESS

1. Scenario Description

A patrolling submarine (searcher) has set up a stationary barrier to intercept transiting submarines (target). Frequently submarines exhibit aspect dependent radiated noise patterns which may become a significant factor in a barrier scenario where the searcher is most frequently presented a bow-on aspect. It is desired to gain an understanding of the magnitude of this problem, and to investigate to what extent the distribution of targets across the barrier will affect intercept performance.

2. PASS Input Data

Table III shows the input data for the base case (B-I).

3. Environment and Search Plan

Figures 3.18 and 3.19 show the propagation loss curves for the searcher and target. Since there are no convergence zones, the MSEL model is not used.

Figure 3.20 shows the search area to be a 90 nm by 200 nm rectangle with the searcher conducting a stationary (back-and-forth) barrier perpendicular to the expected direction of target motion. The placement of the barrier was based on an expected median detection range of 25 nm.

4. Platform Parameters

For the base case (B-I):

1. The searcher speed is constant at 10 knots.
2. The searcher FOM is 87 db and is not target aspect dependent.
3. The initial target distribution across the barrier is uniform.
4. The target starts at the top of the barrier (y=200) (i.e. the searcher is not time late on the barrier).
5. The maximum target course variation is plus-or-minus 30 degrees.
6. The target speed is uniform between 8 and 20 knots.
7. The target changes course and speed (independently) on the average of twice per hour.
8. The target FOM is 84 db.

5. Results and Conclusions

Numerical results for the base case (B-I) and variations of the base case (Case B-II and B-III) are shown in Table IV. The variations in the base case were:

1. Case B-II: The distribution of the initial target position across the top of the barrier was altered so that 50% of the targets were uniformly distributed between 0 and 30 miles from the left boundary, and 50% of the targets were uniformly distributed between 0 and 30 miles from the right boundary.
2. Case B-III: The target distribution is as in B-II. Additionally, the target was assumed to have an aspect dependent source level which induced the target aspect dependent searcher FOM as shown in Table V. The significance of this radiated noise pattern is the bow-null presented the searcher during barrier approach and penetration.

The simulation results indicate that:

1. If the searcher is conducting a constant speed back-and-forth barrier, an increase in the probability of penetration can be realized by transiting the edge of the barrier.
2. When calculating search effectiveness in a barrier scenario, the aspect dependency of target source levels may be a significant factor.

A complete set of graphical output from PASS for Case B-I is shown in Figures 3.21 through 3.25; for Case B-II in Figures 3.26 through 3.30; for Case B-III in Figures 3.31 through 3.39.

TABLE III

INPUT DATA FOR CASE B-I

Search Area Dimensions:

XMAX = 90.00
YMAX = 200.00

Searcher Propagation Loss:

RO(1) = 1.75 DL(1) = 70.00
RO(2) = 3.50 DL(2) = 77.00
RO(3) = 5.25 DL(3) = 80.00
RO(4) = 7.00 DL(4) = 82.00
RO(5) = 11.50 DL(5) = 83.00
RO(6) = 14.00 DL(6) = 87.00
RO(7) = 21.00 DL(7) = 92.00
RO(8) = 28.00 DL(8) = 96.00
RO(9) = 35.00 DL(9) = 100.00

Searcher Track Anchor Points:

NP = 2 KP = 1
XP(1) = 25.00 YP(1) = 25.00
XP(2) = 65.00 YP(2) = 25.00

Target Propagation Loss:

RT(1) = 1.75 TL(1) = 70.00
RT(2) = 3.50 TL(2) = 80.00
RT(3) = 6.00 TL(3) = 93.00
RT(4) = 10.50 TL(4) = 95.00
RT(5) = 16.00 TL(5) = 106.00
RT(6) = 21.00 TL(6) = 114.00
RT(7) = 24.50 TL(7) = 120.00
RT(8) = 26.50 TL(8) = 120.10
RT(9) = 30.00 TL(9) = 127.00

ANG2 = 30.00 START = 0.0

Remaining Platform and Run Parameters:

FOMOD = 87.00 FOMOS = 87.00 SOD = 10.00 SOS = 10.00 TD = 500.00 TS = 500.00
FOMTD = 84.00 FOMTS = 84.00 STMIN = 8.00 STMAX = 20.00 RTSC = 2.00 RTCC = 2.00
SEED = 93000 NREP = 2000 TMAX = 720.00 LAMBDA(1) = 3.00 LAMBDA(2) = 2.00
LAMBDA(3) = 3.00 SIGMA(1) = 3.00 SIGMA(2) = 6.00 SIGMA(3) = 3.00
Compound Error Function Correlation = 0.8000

TABLE IV

NUMERICAL RESULTS FOR CASE B EXAMPLES

	Case B-I	Case B-II	Case B-III
PD	0.7055	0.6040	0.4410
PDDP	1.0000	1.0000	1.0000
MOES	0.8521	0.8014	0.7152
PCD	0.0000	0.0005	0.0075
PCDDP	0.0000	0.0000	1.0000
MOET	0.2926	0.3949	0.5545
ER	2.91	2.03	1.29
(T)s	12.23	12.39	12.80
(T)t	none	none	11.73
(R)s	23.46	24.25	21.40
(R-50)s	22.99	24.04	21.05
(R)t	none	none	5.83
(R-50)t	none	none	4.69

Notes: 1. See Appendix C for explanation of abbreviations.

TABLE V

SEARCHER FOM VS. TARGET ASPECT

Target Aspect	Searcher FOM
000	82
045	87
090	85
135	87
180	82
225	87
270	85
315	87

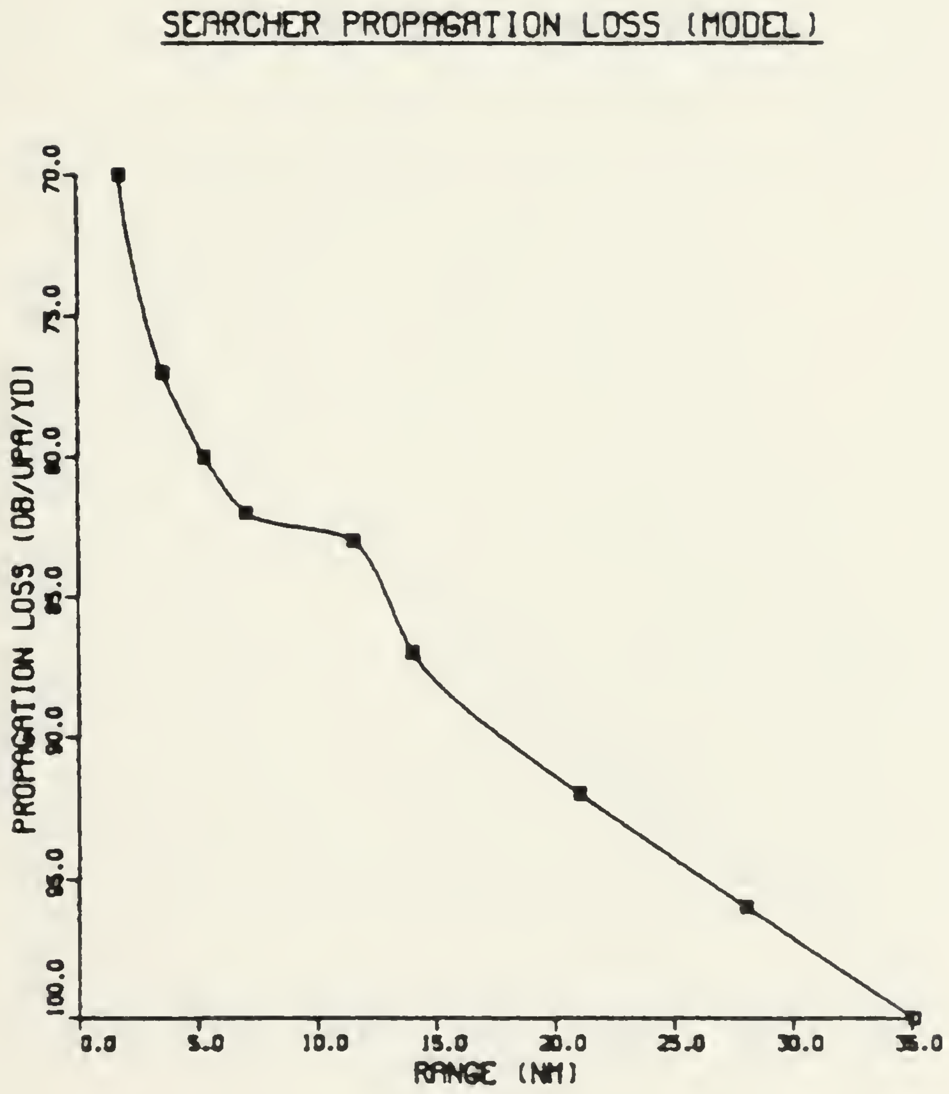

Figure 3.18. Searcher Propagation Loss for Case B Examples.

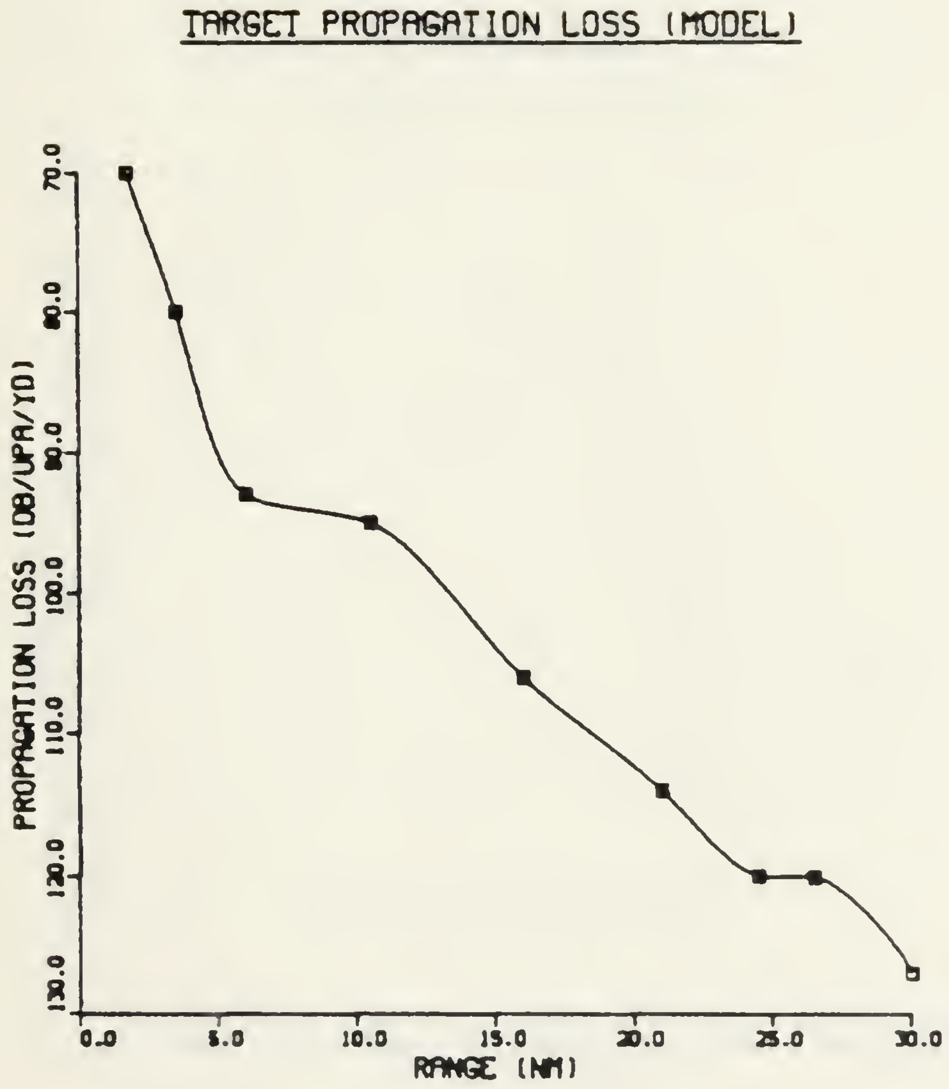

Figure 3.19. Target Propagation Loss for Case B Examples.

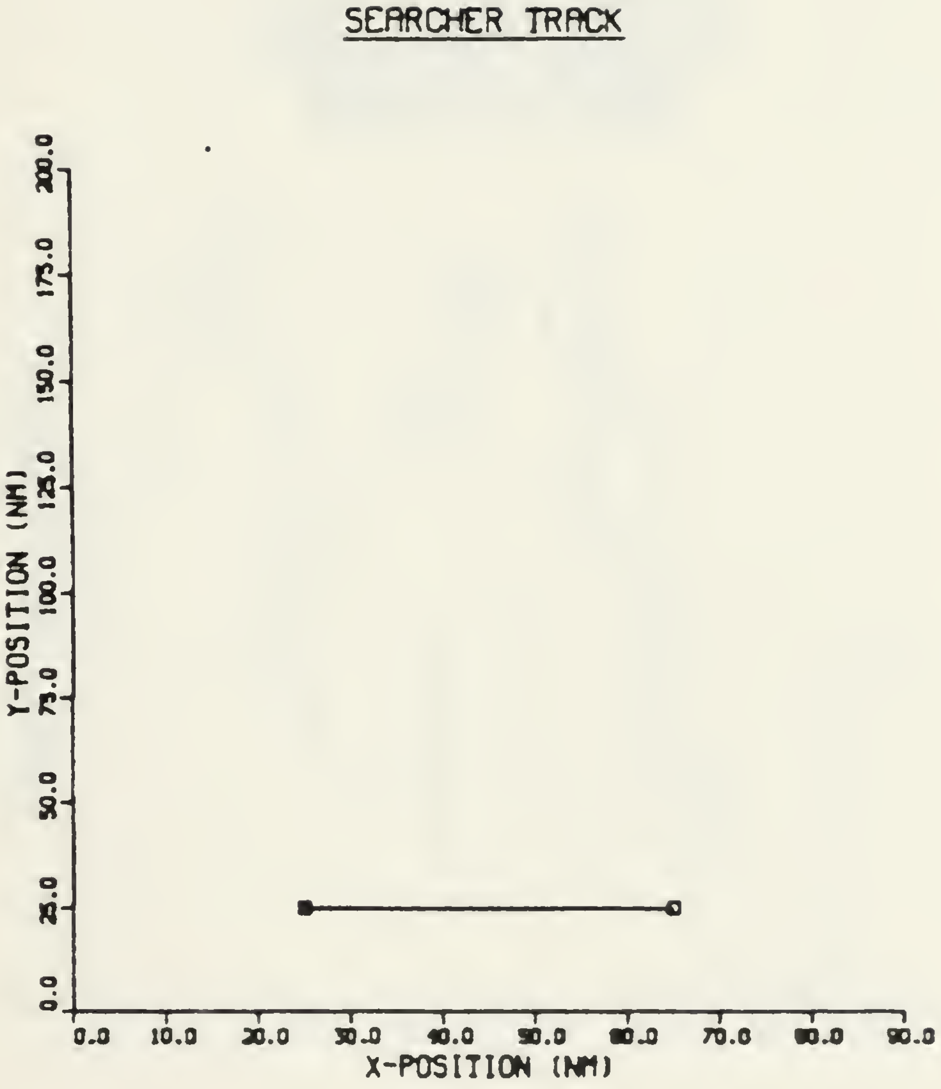

Figure 3.20. Searcher Track for Case B Examples.

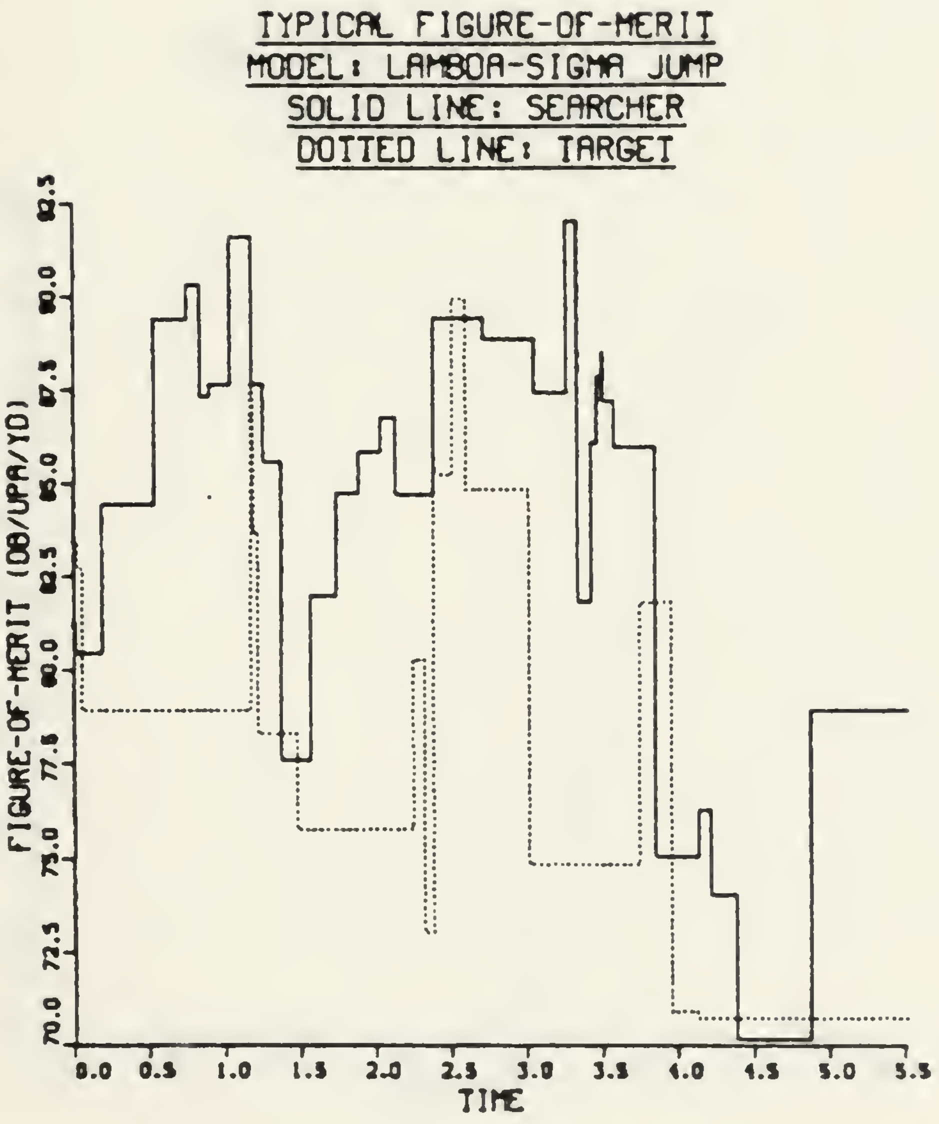

Figure 3.21. Typical Figure-of-Merit for B-I.

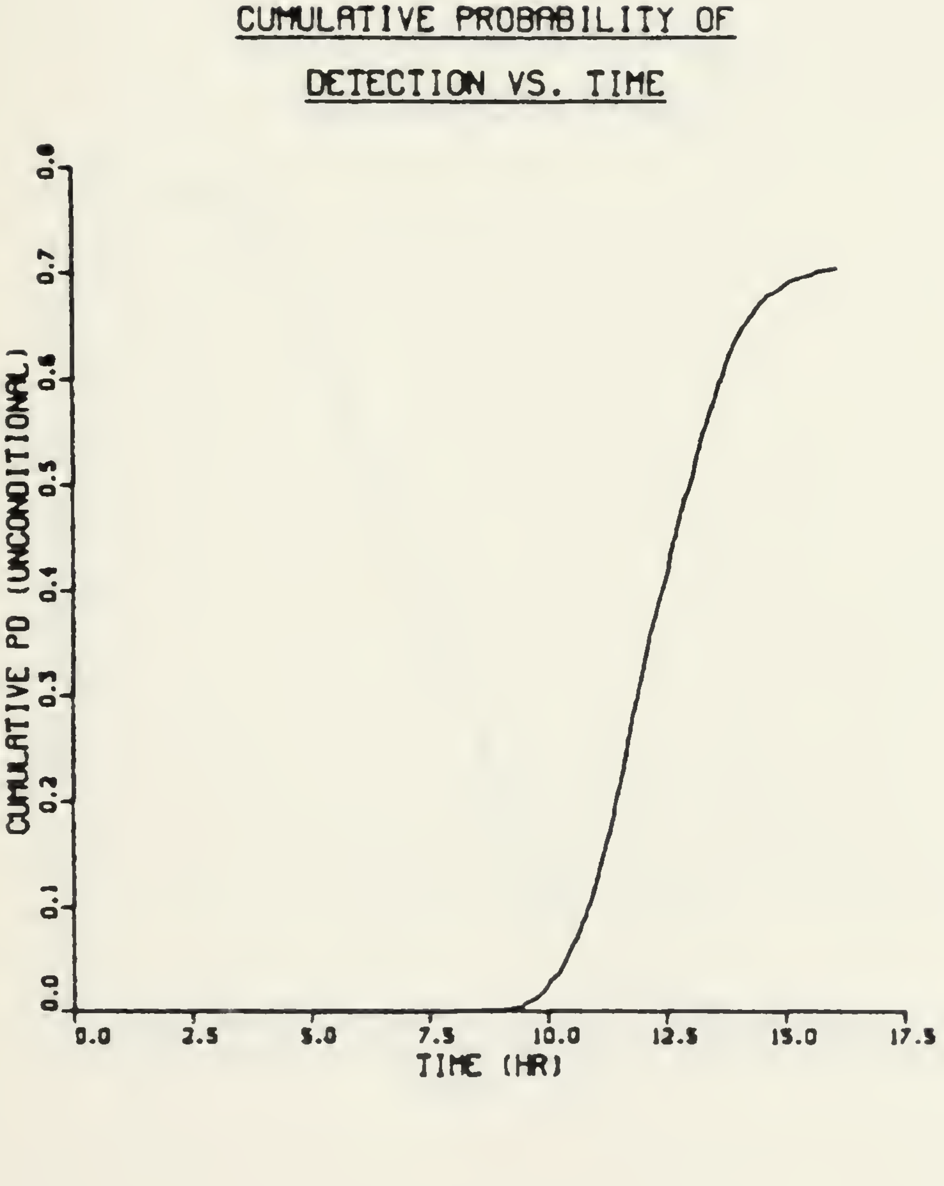

Figure 3.22. Cumulative Probability of Detection for B-I.

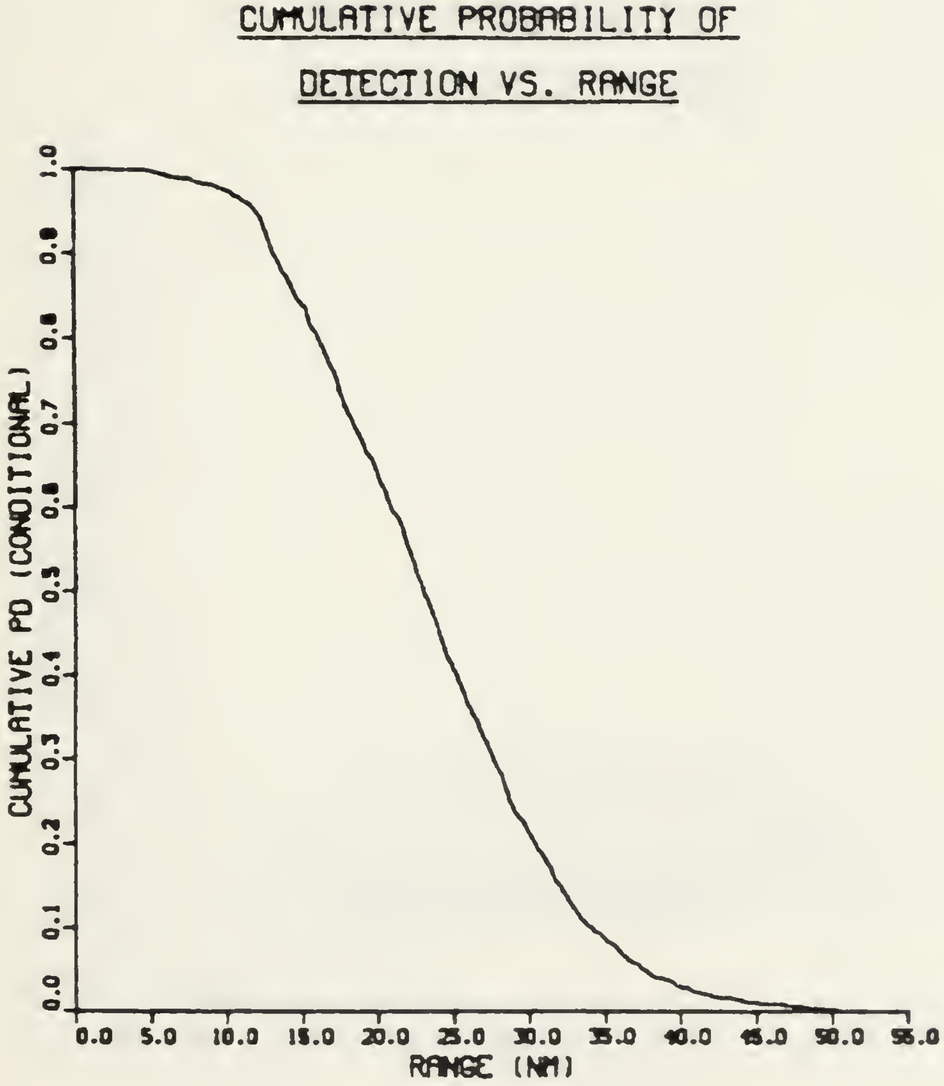

Figure 3.23. Probability of Detection Vs. Range for B-I.

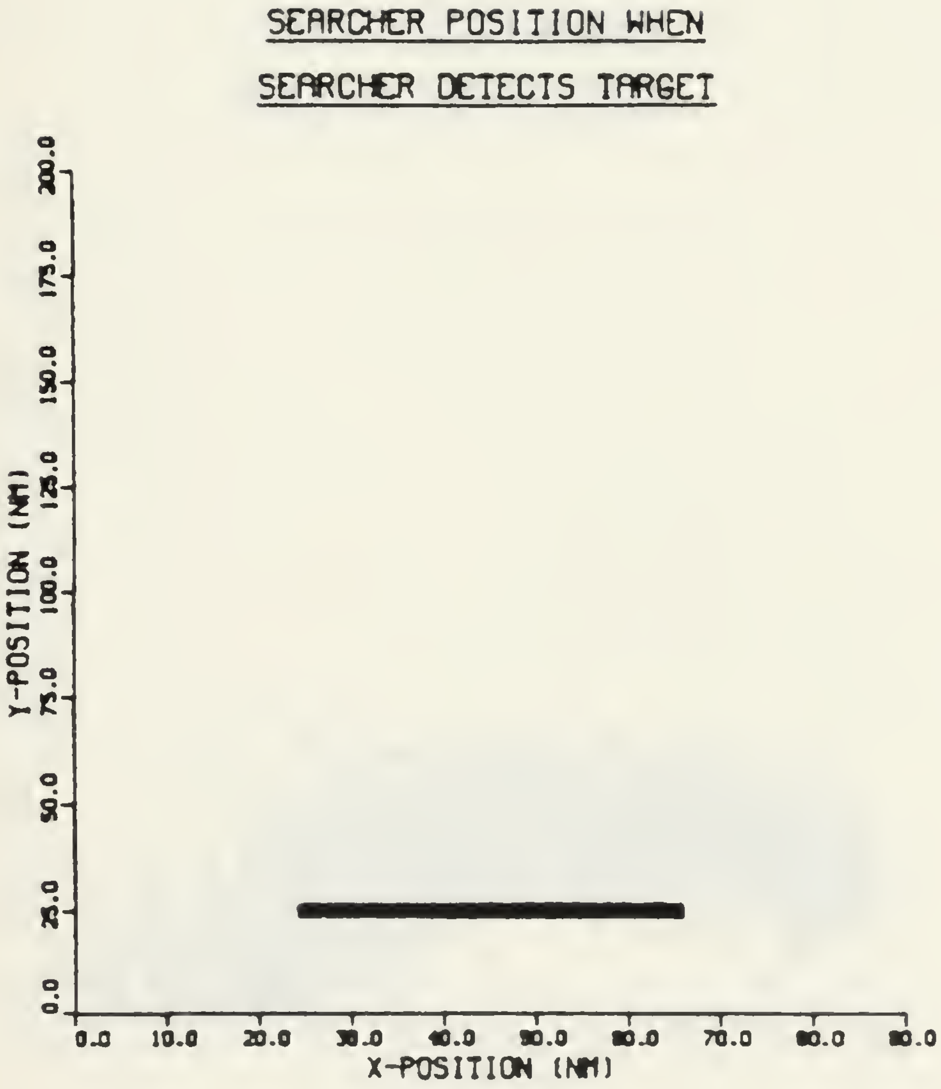

Figure 3.24. Searcher Position at Time of Detection for B-I.

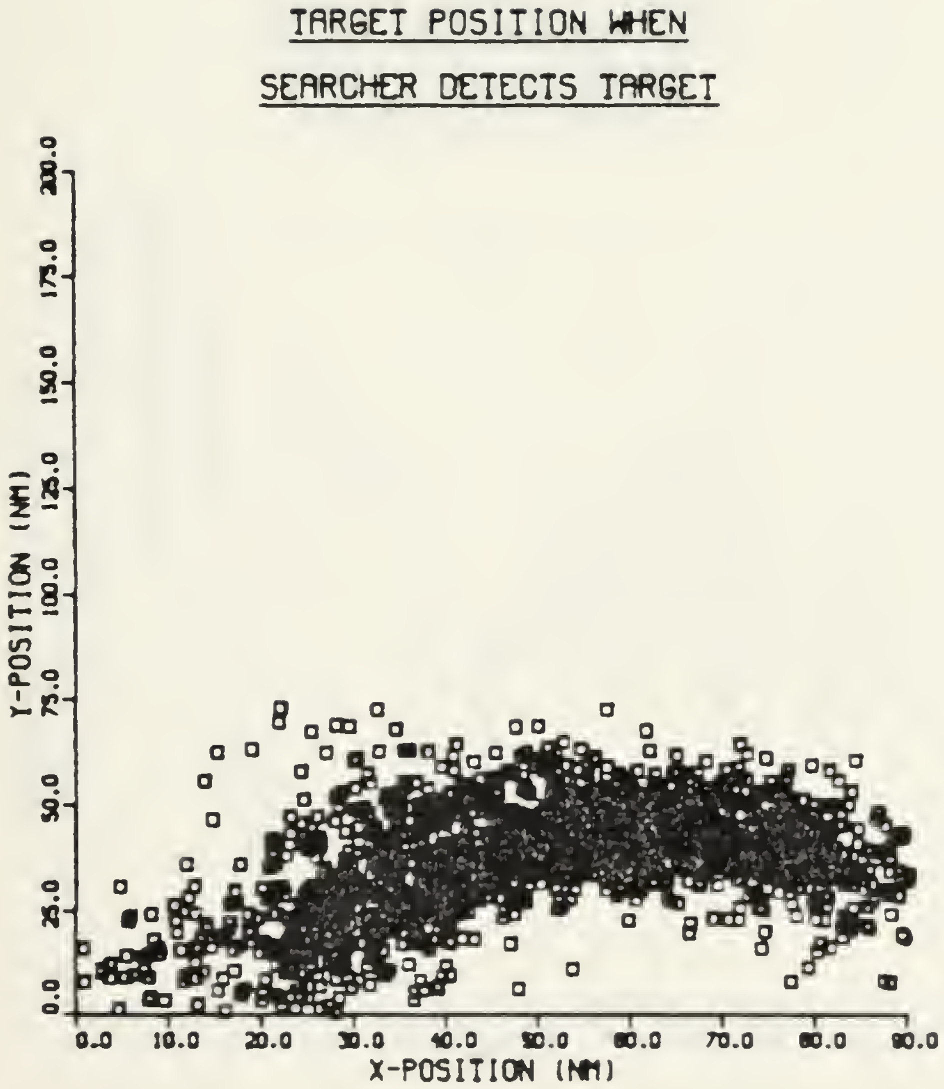

Figure 3.25. Target Position at Time of Detection for B-I.

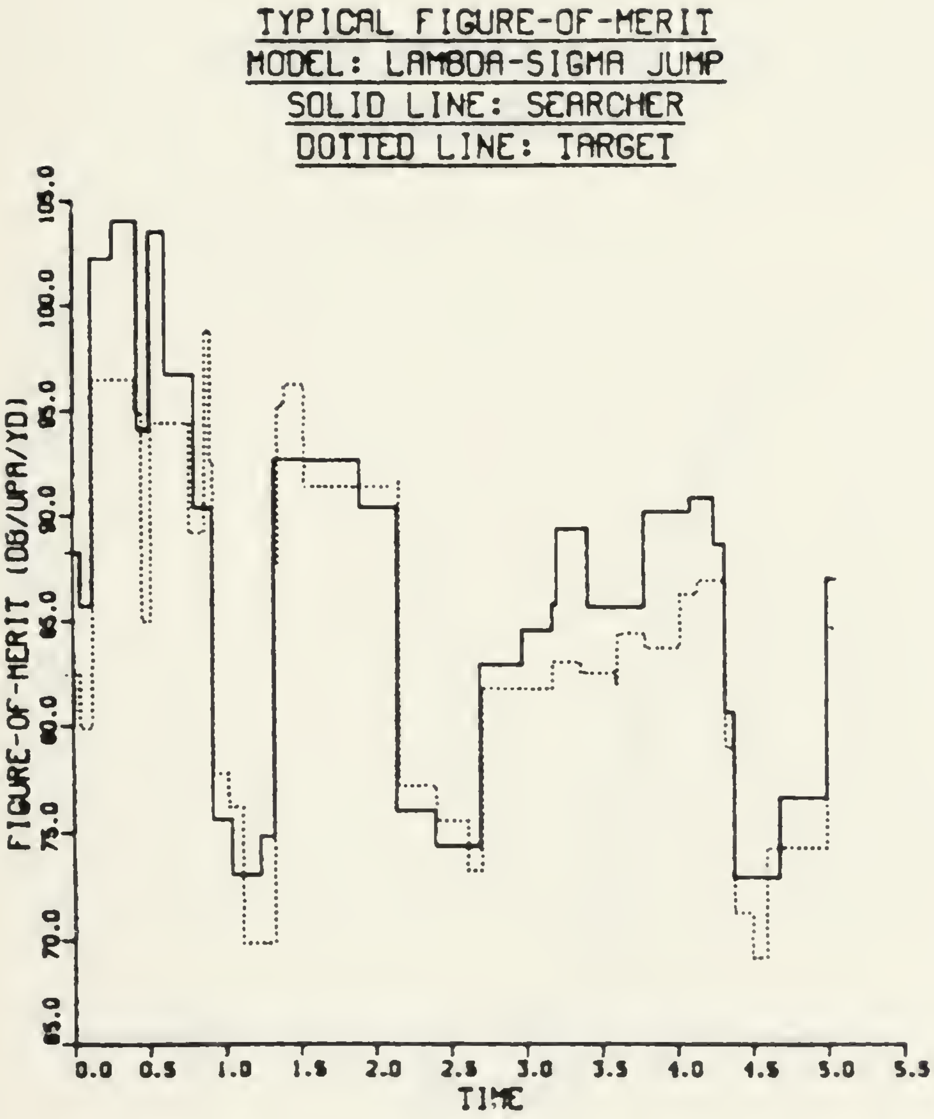

Figure 3.26. Typical Figure-of-Merit for B-II.

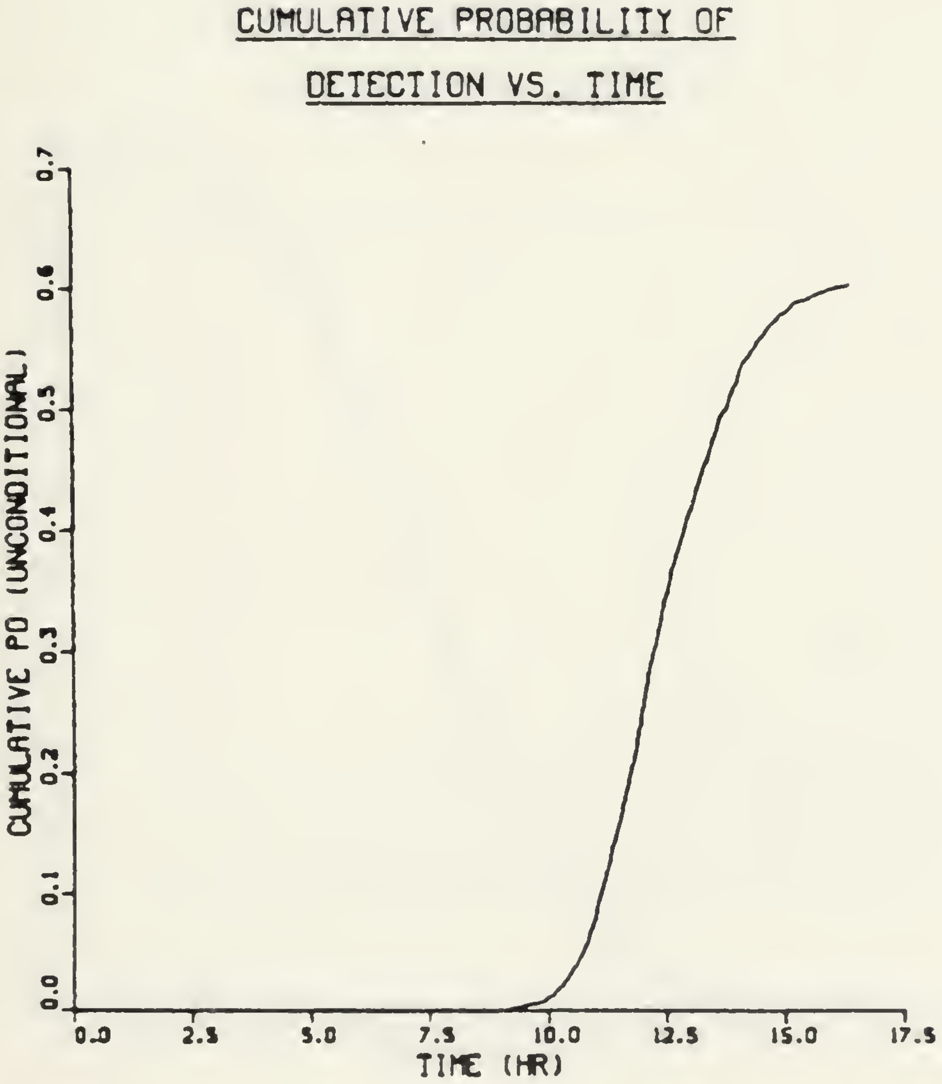

Figure 3.27. Cumulative Probability of Detection for B-II.

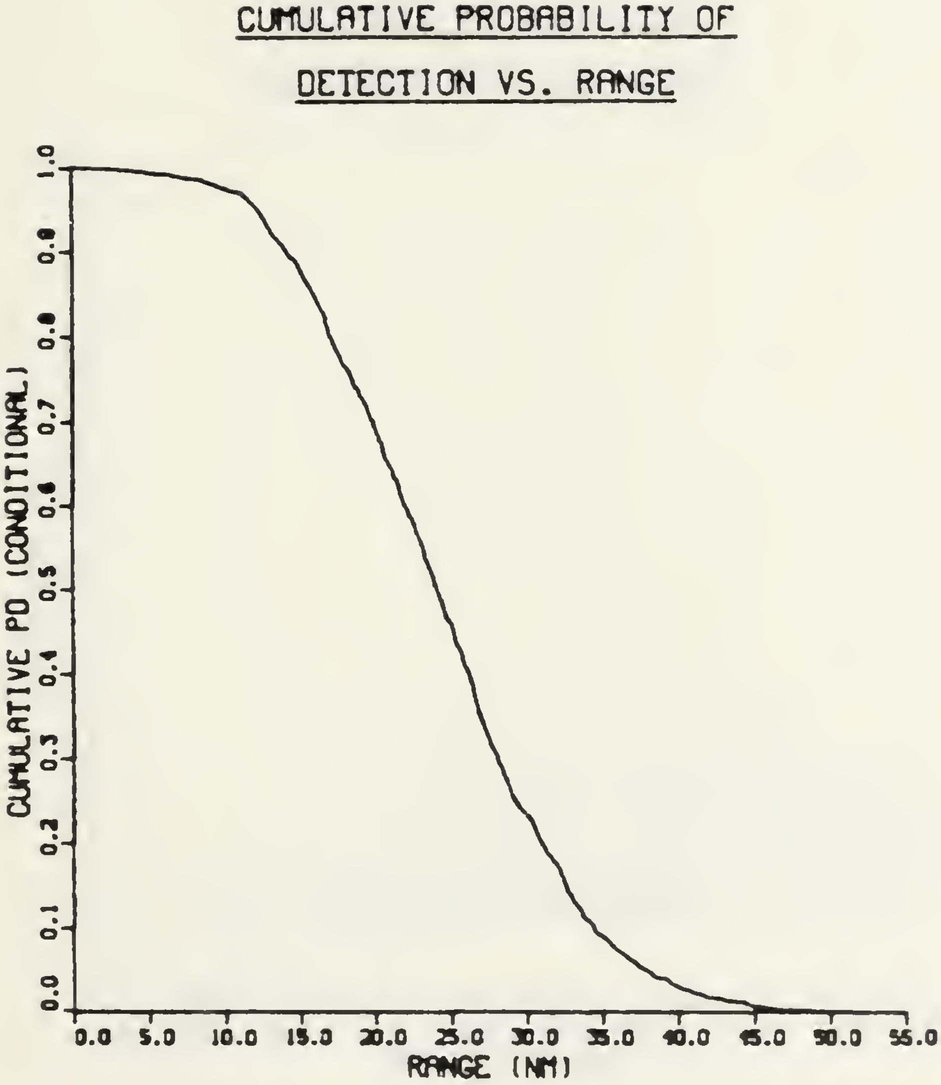

Figure 3.28. Probability of Detection Vs. Range for B-II.

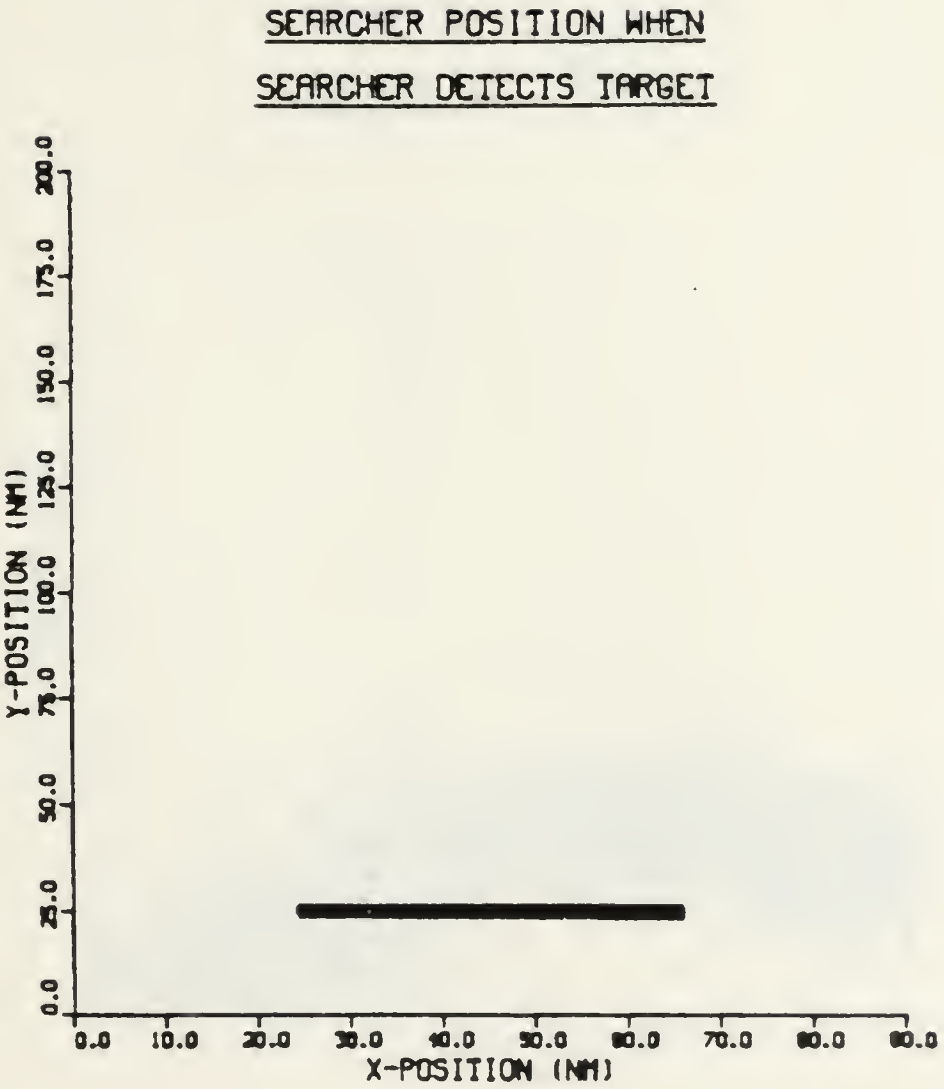

Figure 3.29. Searcher Position at Time of Detection for B-II.

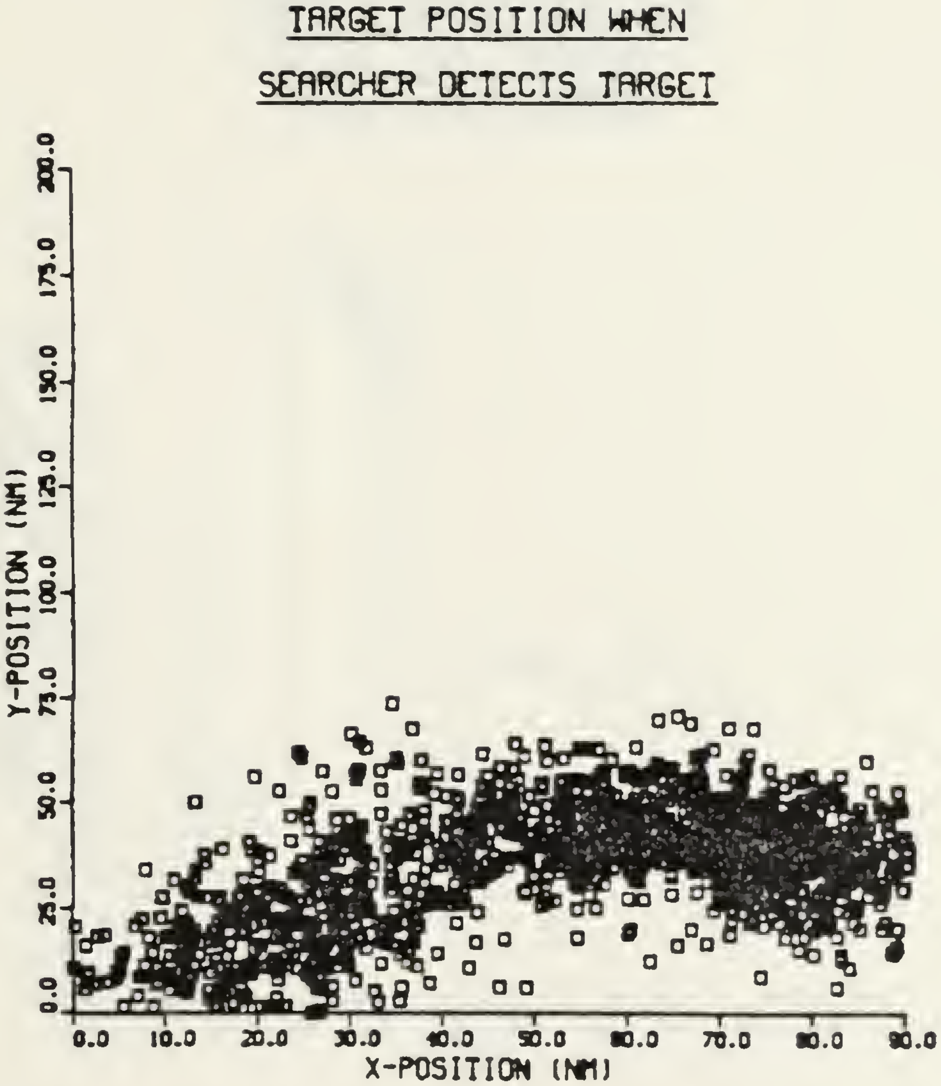

Figure 3.30. Target Position at Time of Detection for B-II.

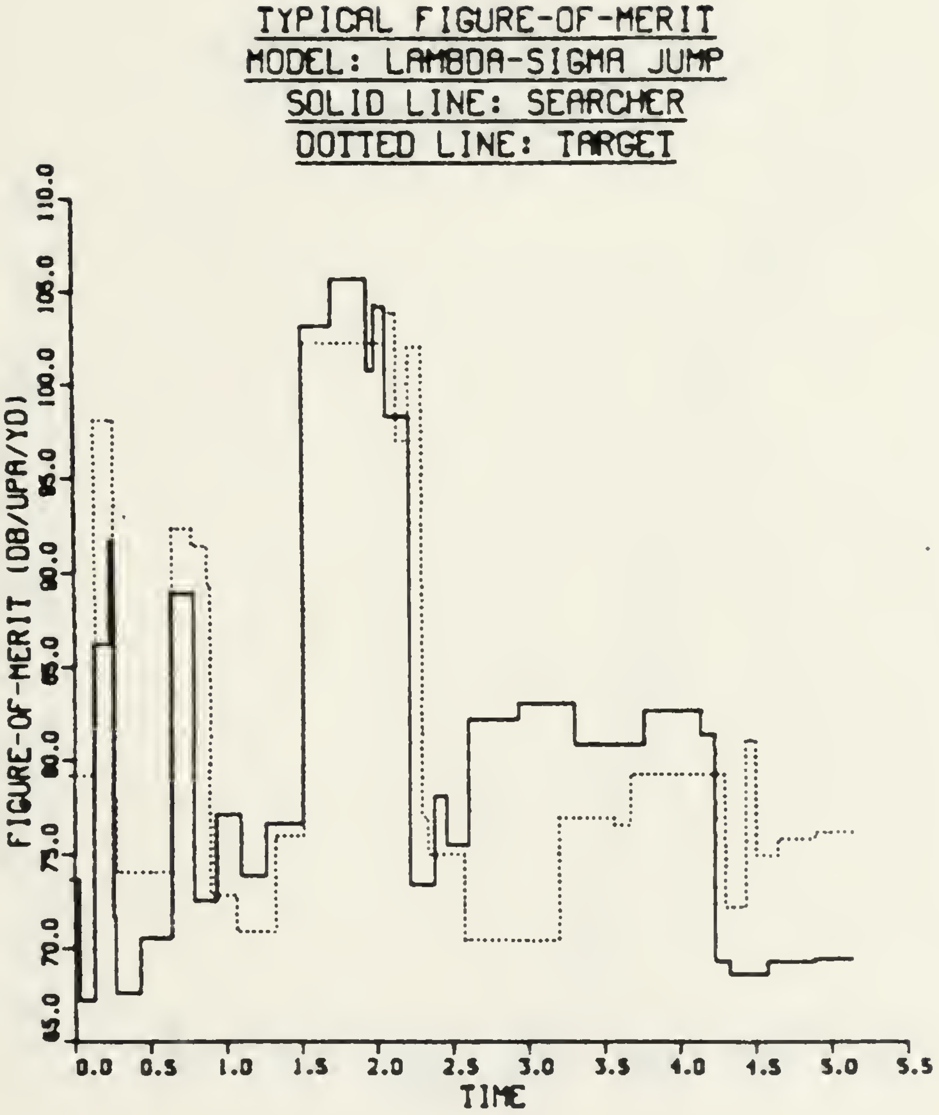

Figure 3.31. Typical Figure-of-Merit for B-III.

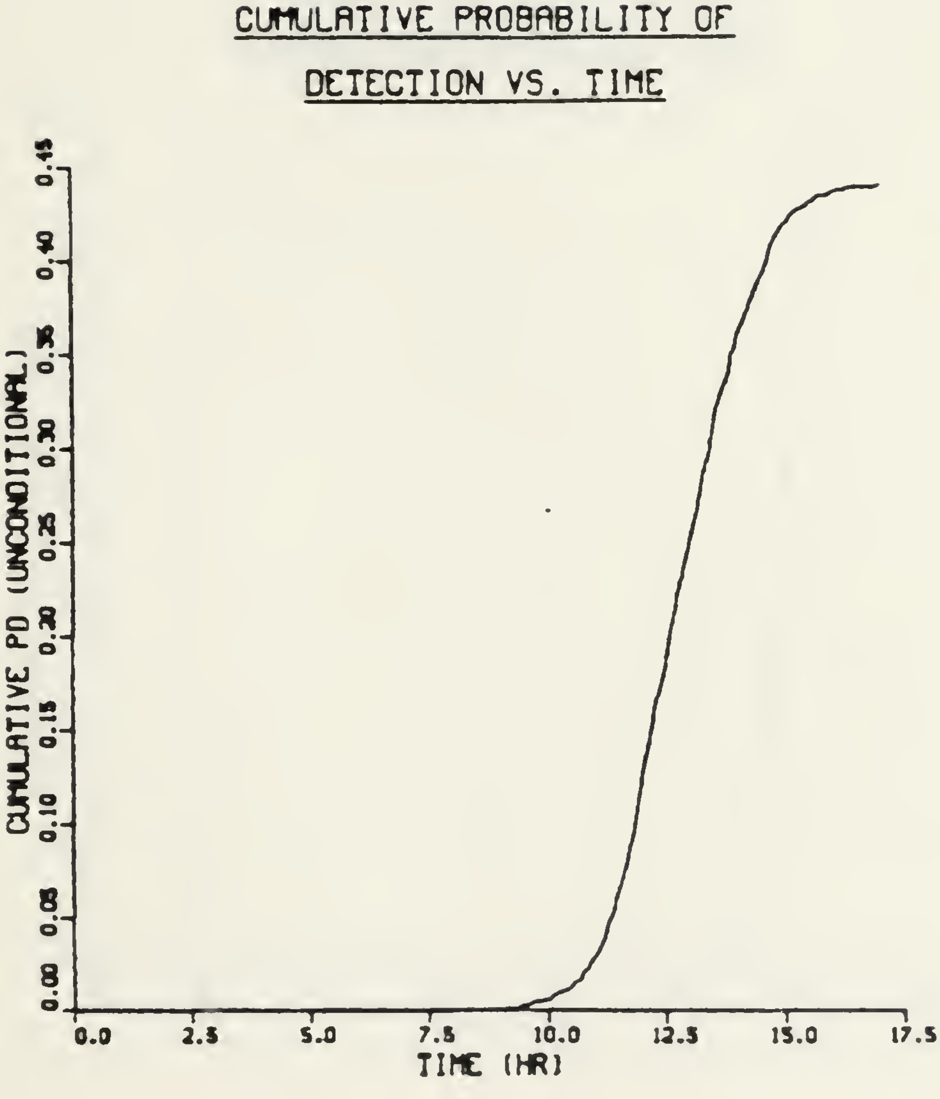

Figure 3.32. Cumulative Probability of Detection for B-III.

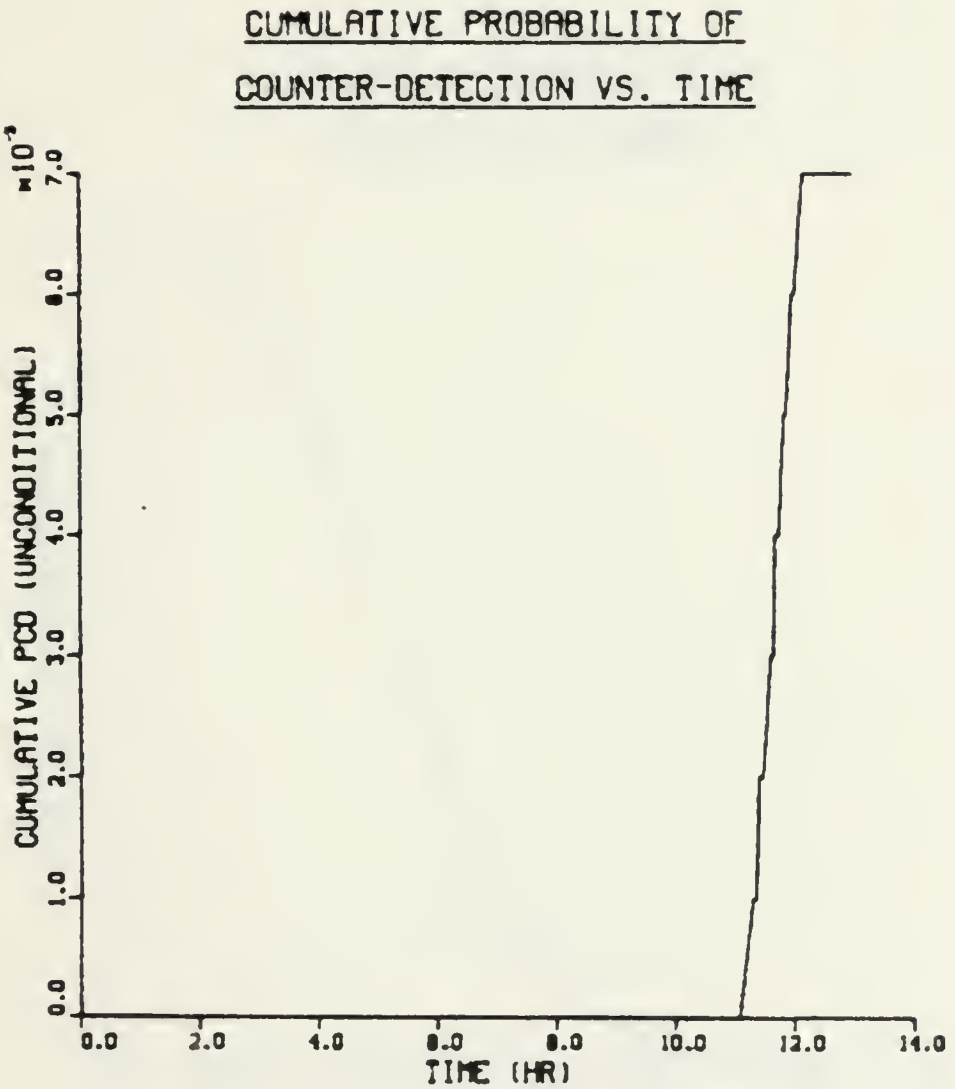

Figure 3.33. Cumulative Probability of Counter-Detection for B-III.

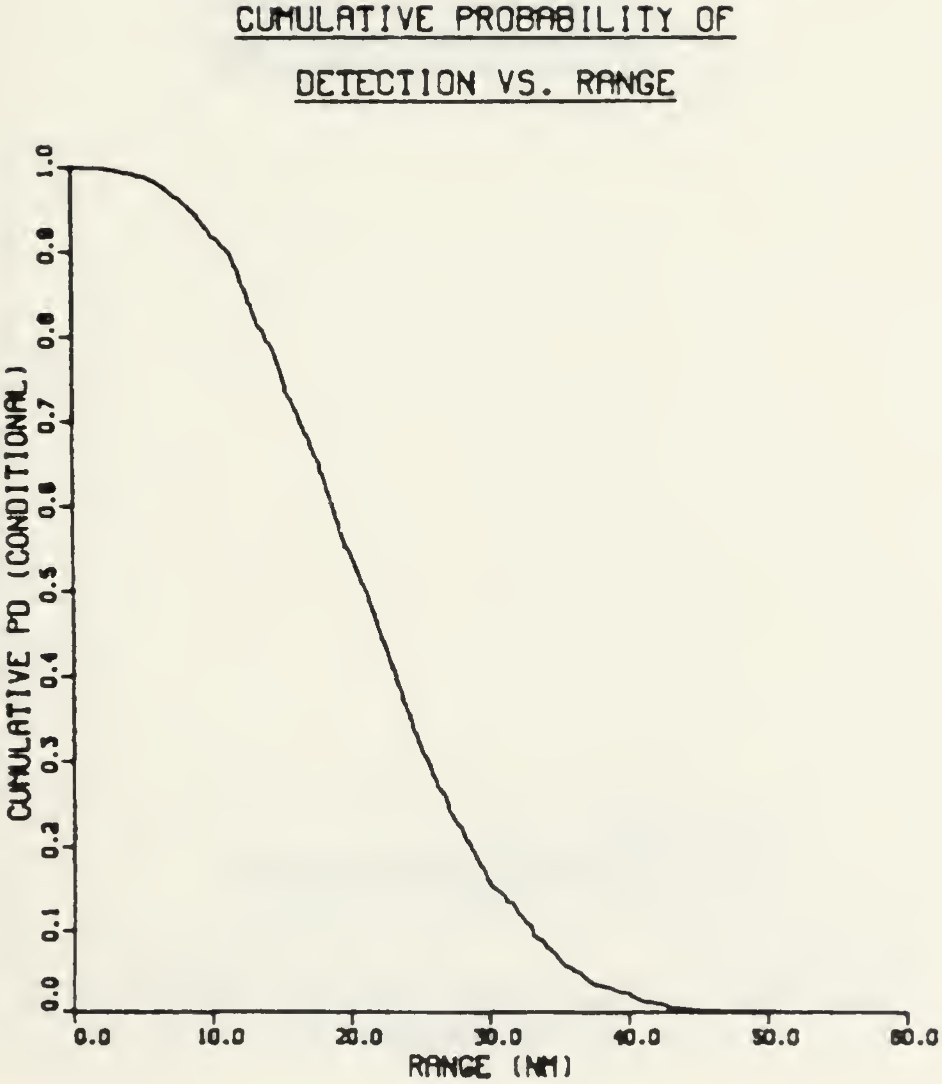

Figure 3.34. Probability of Detection Vs. Range for B-III.

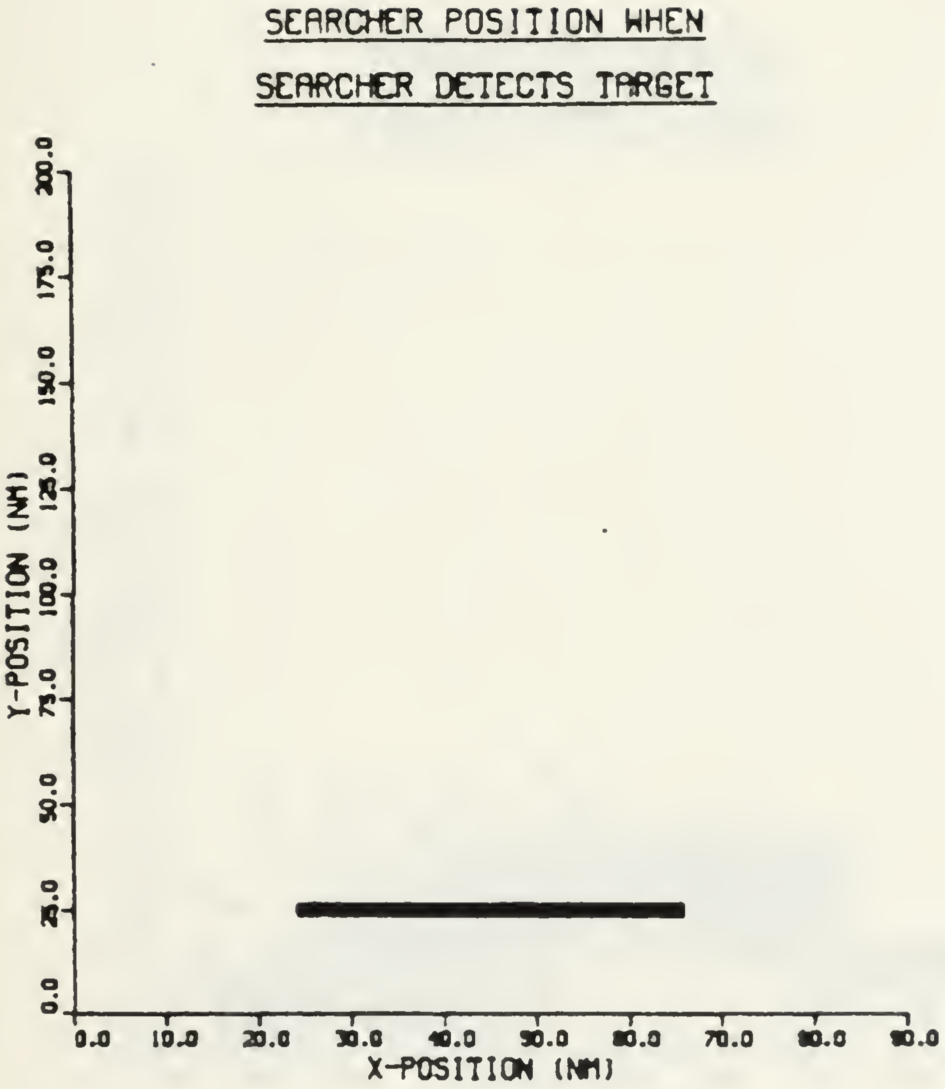

Figure 3.35. Searcher Position at Time of Detection for B-III.

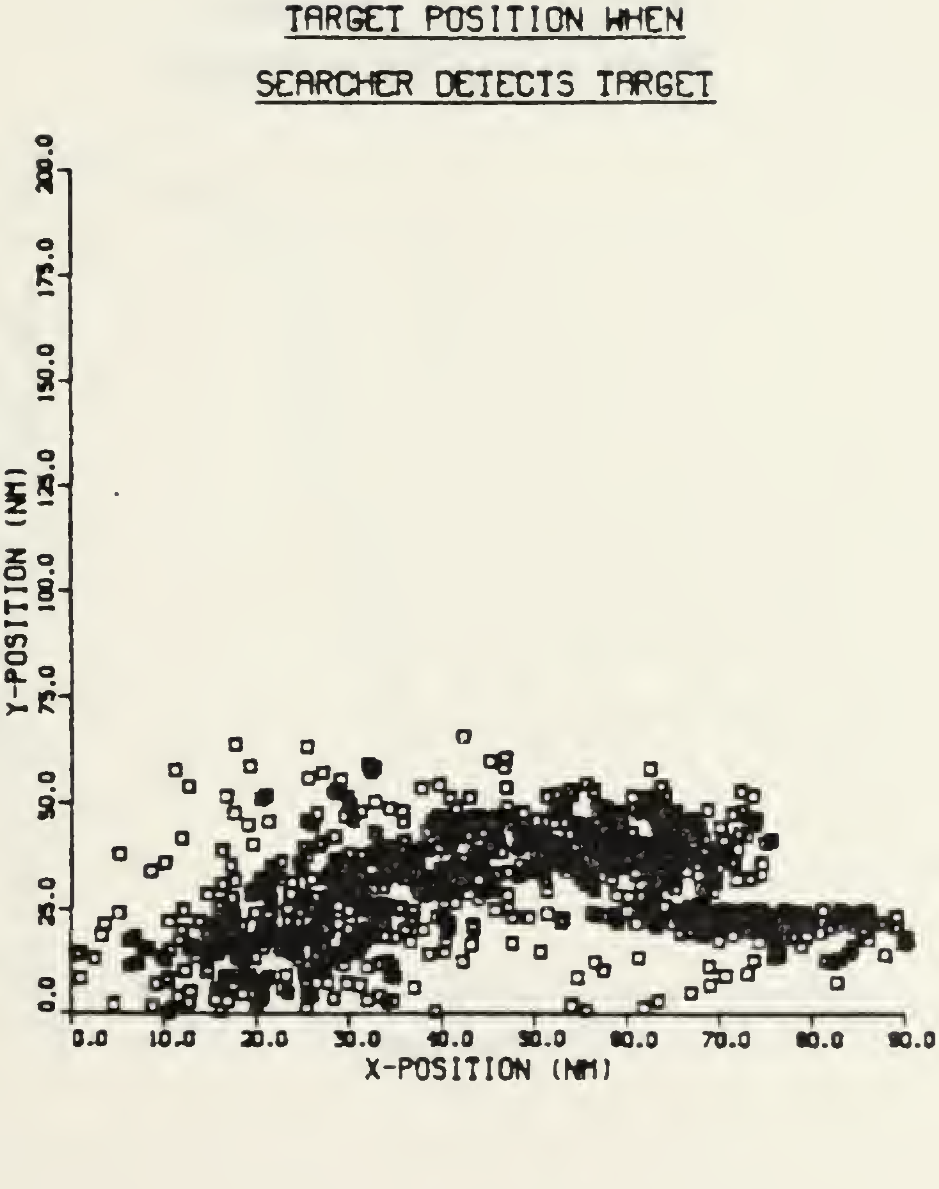

Figure 3.36. Target Position at Time of Detection for B-III.

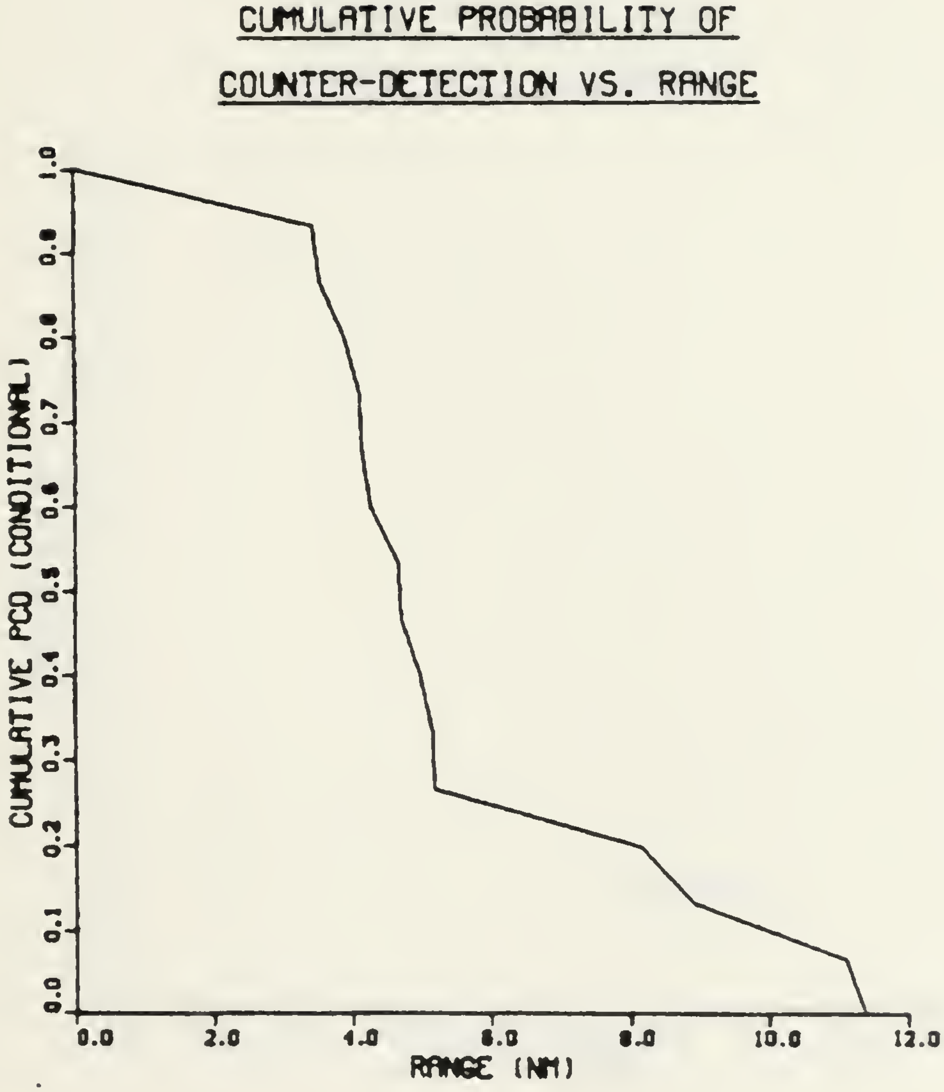

Figure 3.37. Probability of Counter-Dection Vs. Range for B-III.

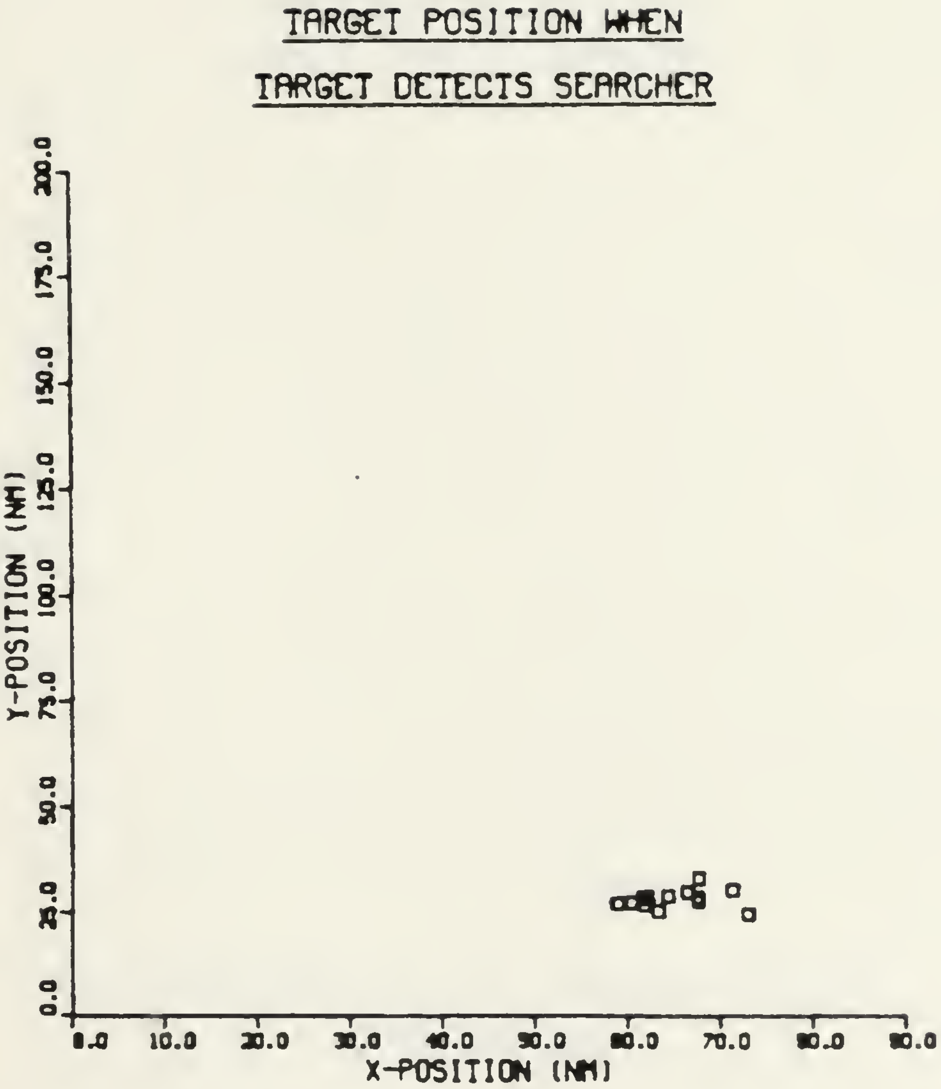

Figure 3.38. Target Position at Time of Counter-Detection for B-III.

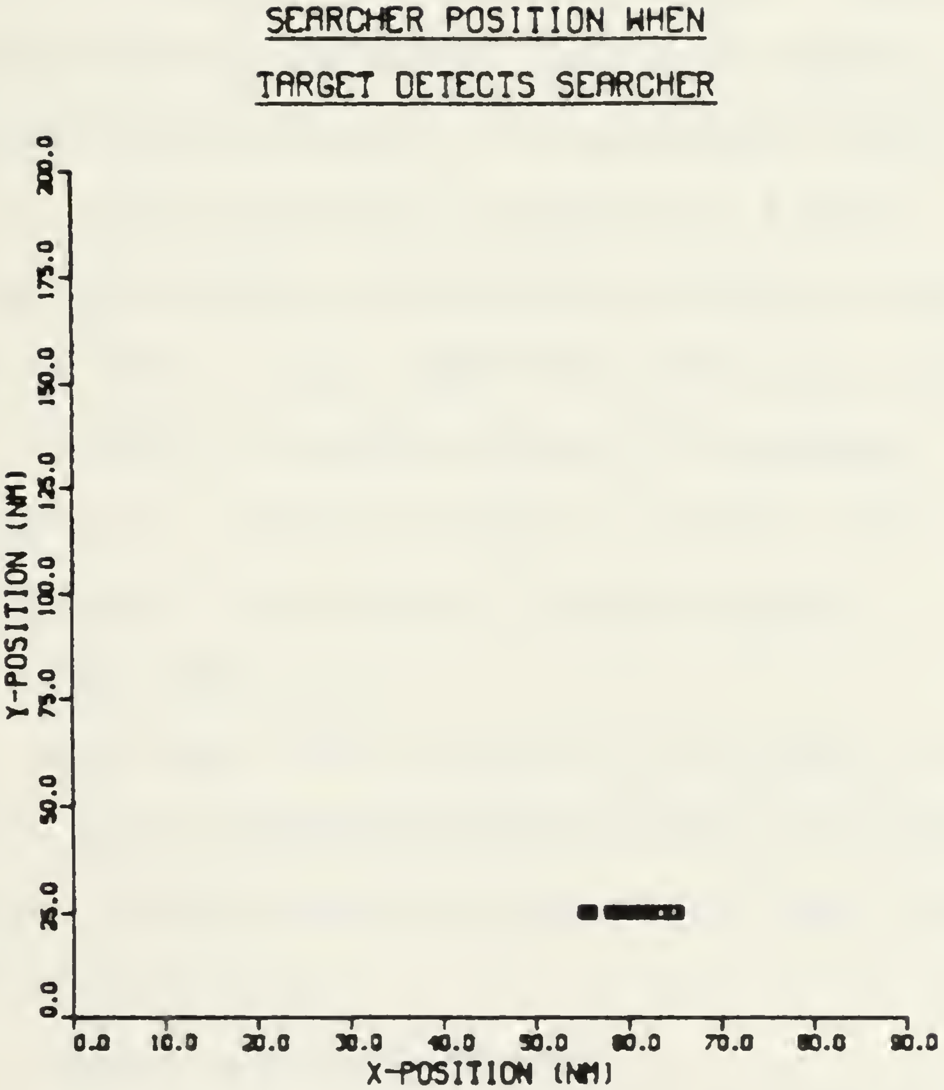

Figure 3.39. Searcher Position at Time of Counter-Detection for B-III.

D. DEFINITE RANGE LAW APPROXIMATIONS

1. Area Search: Best Patrol Speed to Evade a Systematic Searcher

a. Scenario Description

For Case C-I the searcher is given a definite range law sensor (detection range of 20 nm), and the stationary target (uniformly distributed over the search area, A) has no capability to counter-detect. The searcher systematically sweeps the search area at a constant speed (v) with track spacing equal to the detection range. See Figure 3.24. After 88 hours of search (at 5 knots) the searcher has come within one detection range of every point in the search area, or has completely swept the area.

In Case-II target motion is introduced. The target travels at a constant speed of 5 knots, and changes course randomly on the average of twice an hour.

b. PASS Input Data

The input data for Case C-I is shown in Table VI. Note that in these examples figure-of-merit is chosen to give detection ranges based on propagation loss curves shown in Figures 3.40 and 3.41.

c. Results and Conclusions

Numerical results for Case C-I and Case C-II are shown in Table VII. Graphical results for Case C-I are shown in Figures 3.43 through 3.46, and for Case C-II in Figures 3.47 through 3.49.

A comparison of the results shows:

1. The times to detection are roughly uniform for Case C-I as can be seen in the very nearly constant slope of Figure 3.43. Note the departure from exhaustive search results (manifested by the jump at time = 0 in Figure 3.43), which is due to the significant probability of detection at the instant the search starts. Also, "bumps" in the cumulative probability of detection curve occur at turn points, and represent the combined effects of the searcher having a significant portion of his detection circle outside the search area, and sweeping area previously swept, thus resulting in a low detection rate.

2. Introducing target motion in Case C-II skews the distribution of time to detection to the right (see Figure 3.47), resulting in an increased mean time to detection.

Conclusion 2 above raises the question: What is the "best" speed to evade? The results of Case C-II indicate that the best speed is not zero, if the criteria is mean time to detection. Before proceeding, the following terms are defined:

1. Speed Ratio: The ratio of target speed (u) to searcher speed (v). Note that if the speed ratio is zero, then the target is stationary.

2. Mean Time Ratio: The ratio of the mean time to detection (for a given speed ratio) to the mean time of detection when the speed ratio is zero.

3. Median Time Ratio: The ratio of the median time to detection (for a given speed ratio) to the median time to detection when the speed ratio is zero.

To investigate the problem of best evasion speed, PASS was run for various search speeds (v = 5, 10, 15, 20 knots) and various speed ratios at each search speed, under case C conditions (definite range law sensor).

Figures 3.50 and 3.51 show the results of these simulations. Note that whenever the time ratio is greater than 1.0, the time to detection is greater than the time to detection for a stationary target. Figure 3.50 shows the mean time to detection peaks at speed ratios between 1.0 and 1.4, depending on the searcher speed. Figure 3.51 shows the median time to detection is relatively insensitive to speed ratios below 1.0, and drops off rather sharply for speed ratios above 1.0. The conclusion drawn is that the best speed to evade the searcher is about equal to the searcher speed. If the searcher speed is not known, the best tactic would be to limit maximum patrol speed to the most likely minimum searcher speed. It is important to note that the best patrol speed is not necessarily the minimum achievable speed.

TABLE VI

INPUT DATA FOR CASE C-I

Search Area Dimensions:

XMAX = 120.00
YMAX = 120.00

Searcher Track Anchor Points:

NP = 6   KP = 1
XP(1) = 100.00   YP(1) = 0.0
XP(2) = 100.00   YP(2) = 120.00
XP(3) = 60.00   YP(3) = 120.00
XP(4) = 60.00   YP(4) = 0.0
XP(5) = 20.00   YP(5) = 0.0
XP(6) = 20.00   YP(6) = 120.00

Searcher Propagation Loss:

RO(1) = 2.50   OL(1) = 70.00
RO(2) = 5.00   OL(2) = 85.00
RO(3) = 7.50   OL(3) = 92.90
RO(4) = 10.00   OL(4) = 93.10
RO(5) = 20.00   OL(5) = 94.00
RO(6) = 30.00   OL(6) = 96.00
RO(7) = 45.00   OL(7) = 104.00
RO(8) = 60.00   OL(8) = 108.00
RO(9) = 85.00   OL(9) = 116.00
RO(10) = 95.00   OL(10) = 120.00

Target Propagation Loss:

RT(1) = 2.50   TL(1) = 70.00
RT(2) = 5.00   TL(2) = 95.00
RT(3) = 7.50   TL(3) = 104.00
RT(4) = 10.00   TL(4) = 105.00
RT(5) = 20.00   TL(5) = 109.00
RT(6) = 30.00   TL(6) = 115.00
RT(7) = 45.00   TL(7) = 130.00
RT(8) = 60.00   TL(8) = 136.00
RT(9) = 85.00   TL(9) = 146.00
RT(10) = 95.00   TL(10) = 150.00

Remaining Platform and Run Parameters:

FOMOD = 94.00   FOMOS = 94.00   SOD = 5.00   SOS = 5.00   TD = 100.00   TS = 100.00
FOMTD = 70.00   FOMTS = 70.00   STMIN = 0.0   STMAX = 0.0   RTSC = 0.001   RTCC = 2.00
SEED = 92235   NREP = 5000   TMAX = 88.00   LAMBDA(1) = 0.001   LAMBDA(2) = 0.001
LAMBDA(3) = 0.001   SIGMA(1) = 0.0   SIGMA(2) = 0.0   SIGMA(3) = 0.0

TABLE VII

NUMERICAL RESULTS FOR CASES C-I AND C-II

	Case C-I	Case C-II
PD	1.0000	0.9992
PDDP	1.0000	1.0000
MOES	1.0000	0.9994
PCD	0.0000	0.0000
PCDDP	0.0000	0.0000
MOET	0.0000	0.0002
ER	infinite	4496.00
(T)s	40.12	43.84
Var(T)s	707.17	1195.38
T(min)	0.00	0.00
T(max)	87.24	302.07

Notes: 1. See Appendix C for explanation of abbreviations.

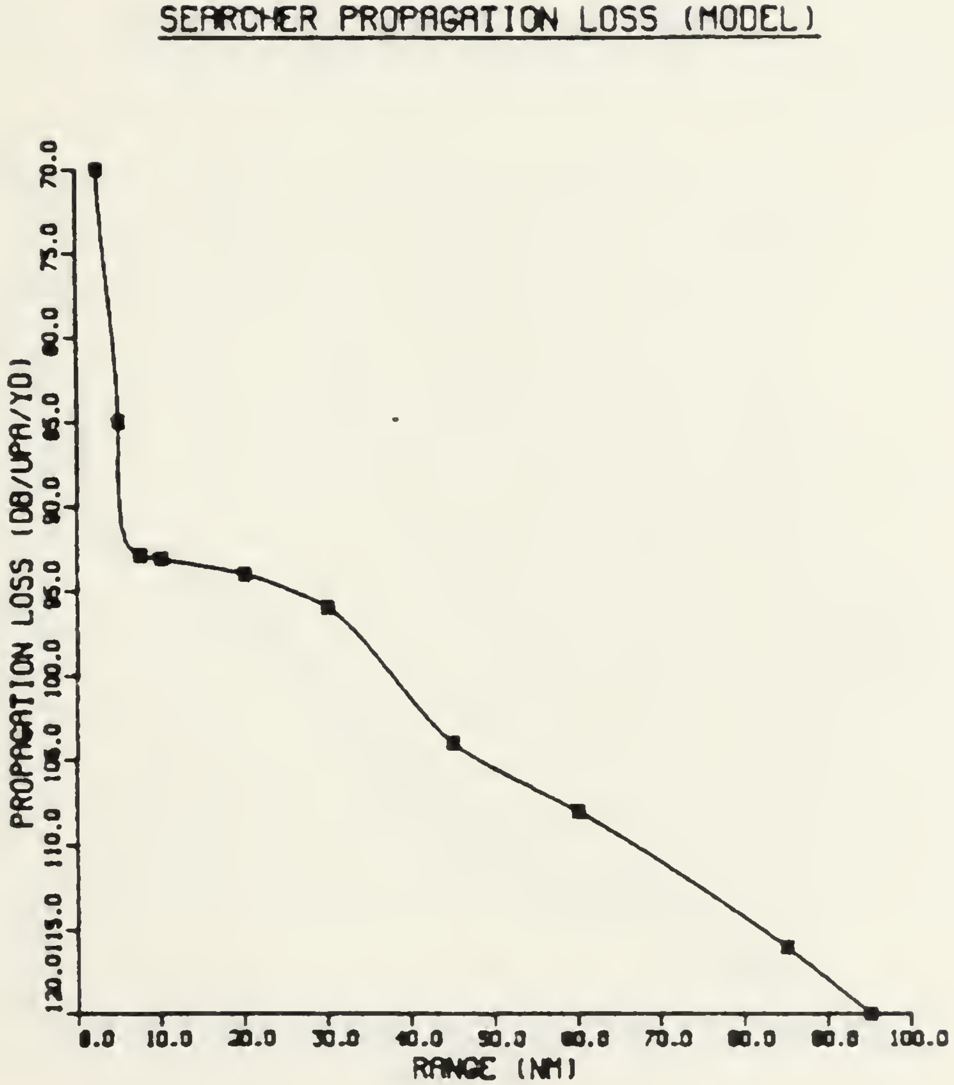

Figure 3.40. Searcher Propagation Loss for Case C Examples.

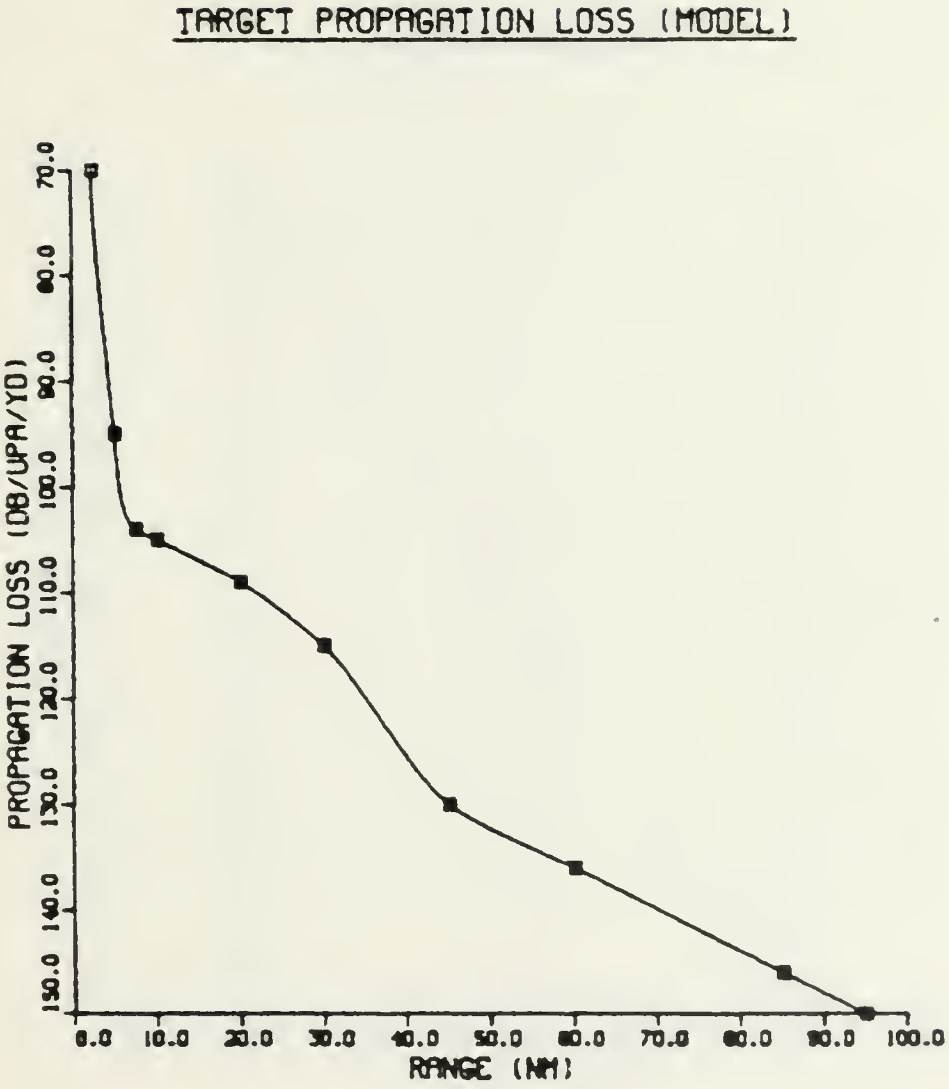

Figure 3.41. Target Propagation Loss for Case C Examples.

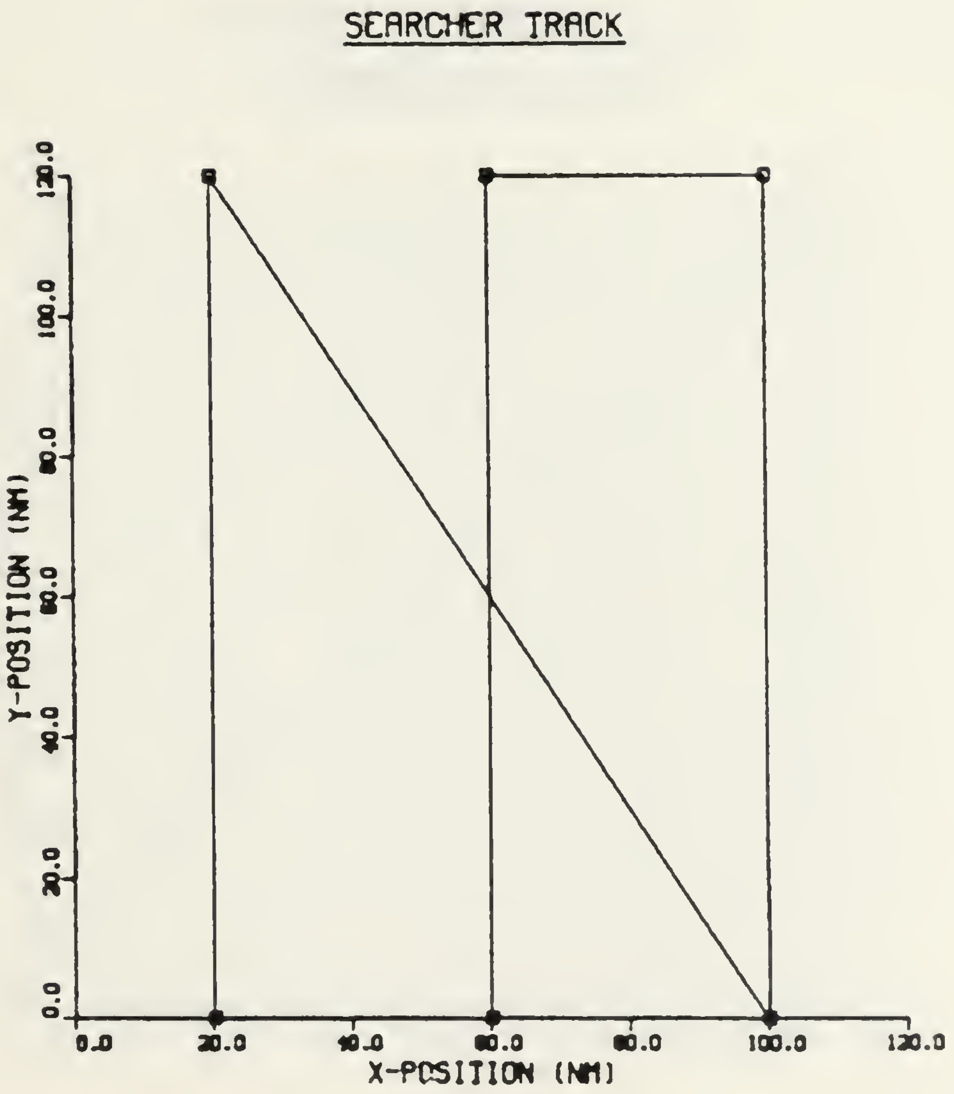

Figure 3.42. Searcher Track for Cases C-I and C-II.

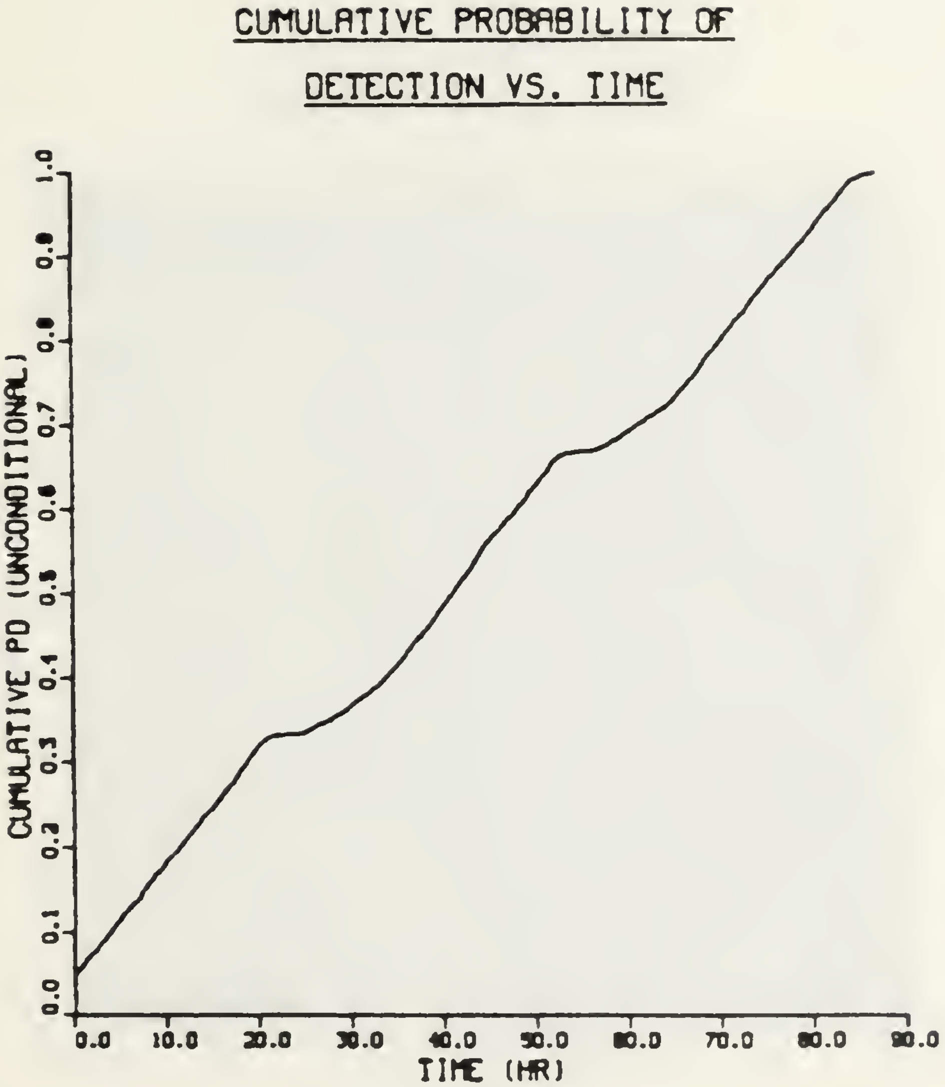

Figure 3.43. Cumulative Probability of Detection for C-I.

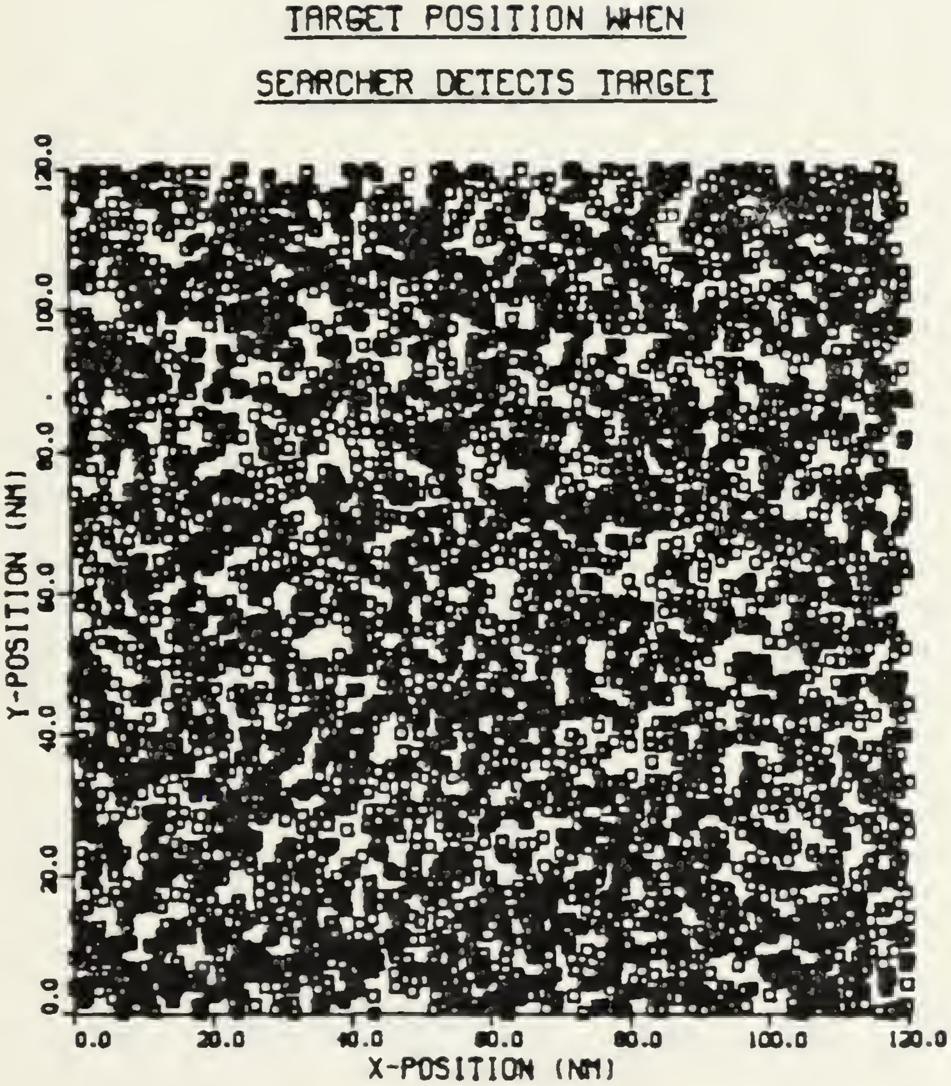

Figure 3.44. Target Position at Time of Detection for C-I.

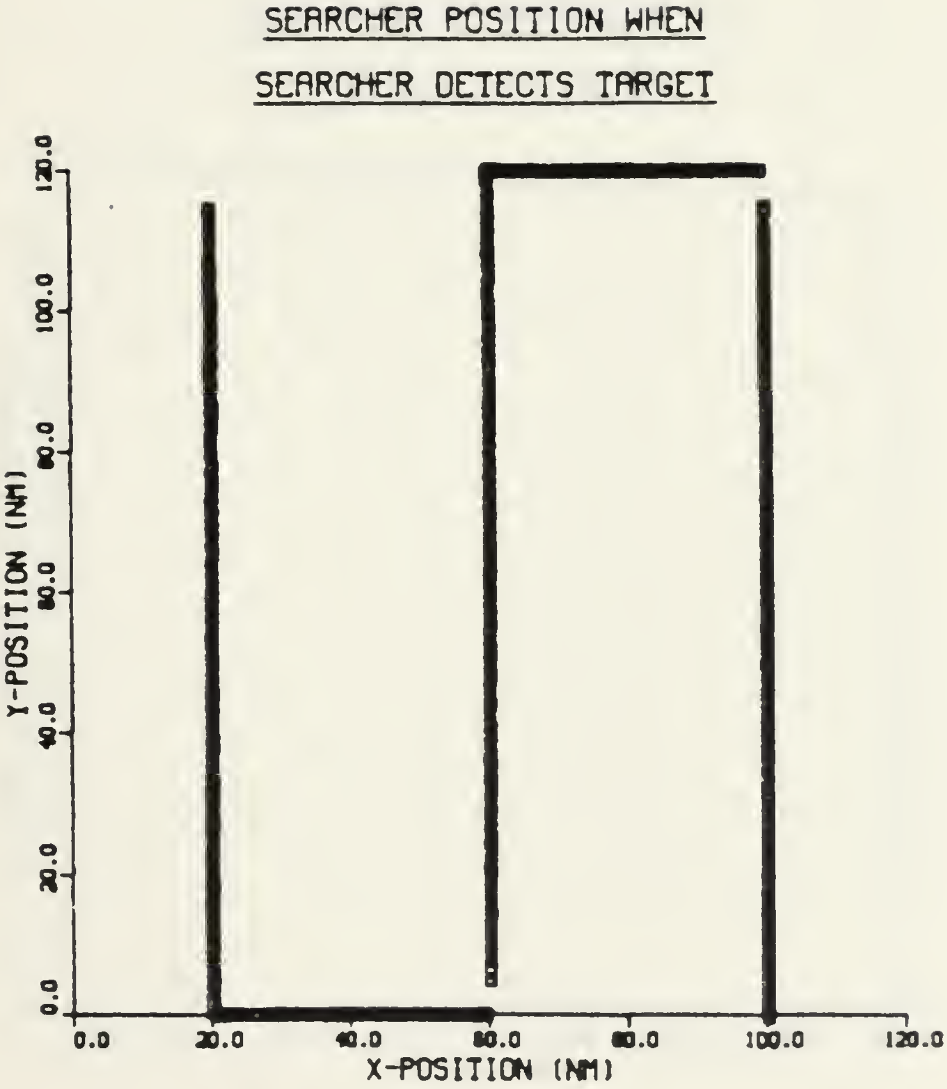

Figure 3.45. Searcher Position at Time of Detection for C-I.

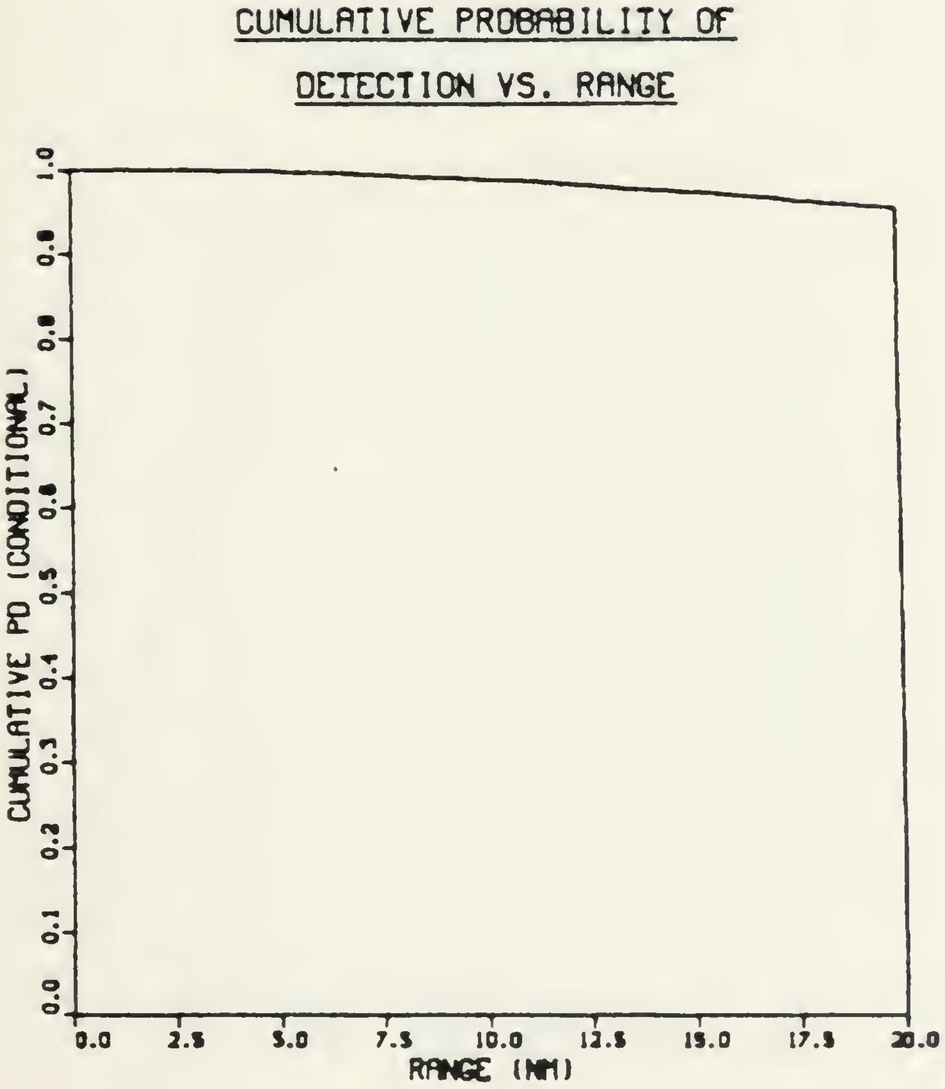

Figure 3.46. Probability of Detection Vs. Range for C-I.

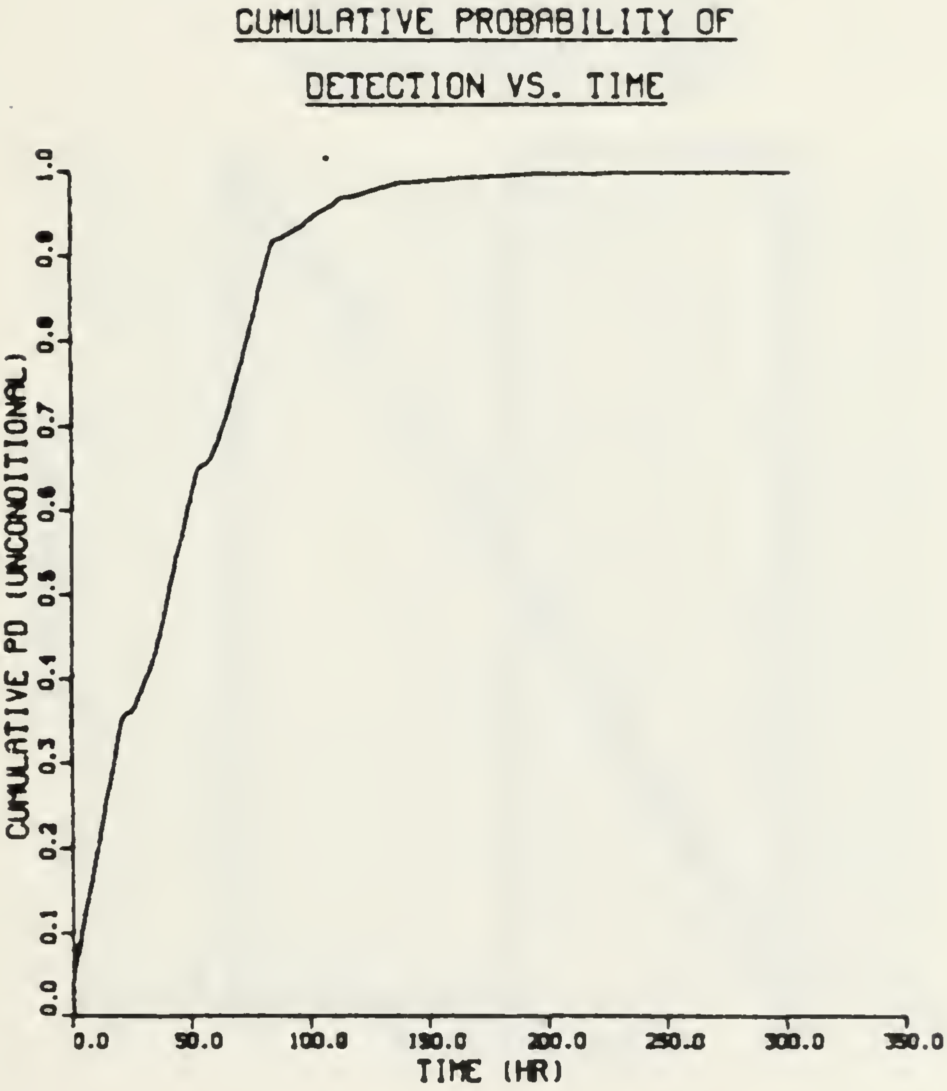

Figure 3.47. Cumulative Probability of Detection for C-II.

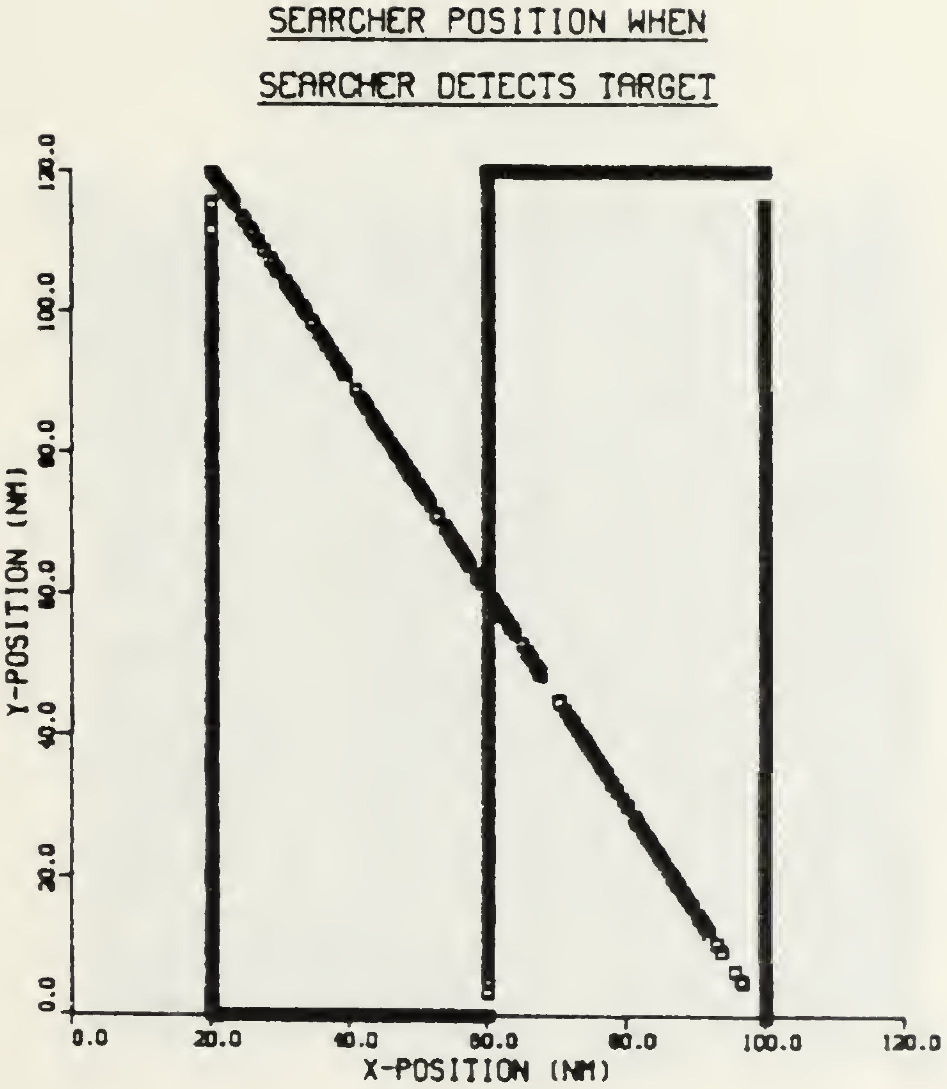

Figure 3.48. Searcher Position at Time of Detection for C-II.

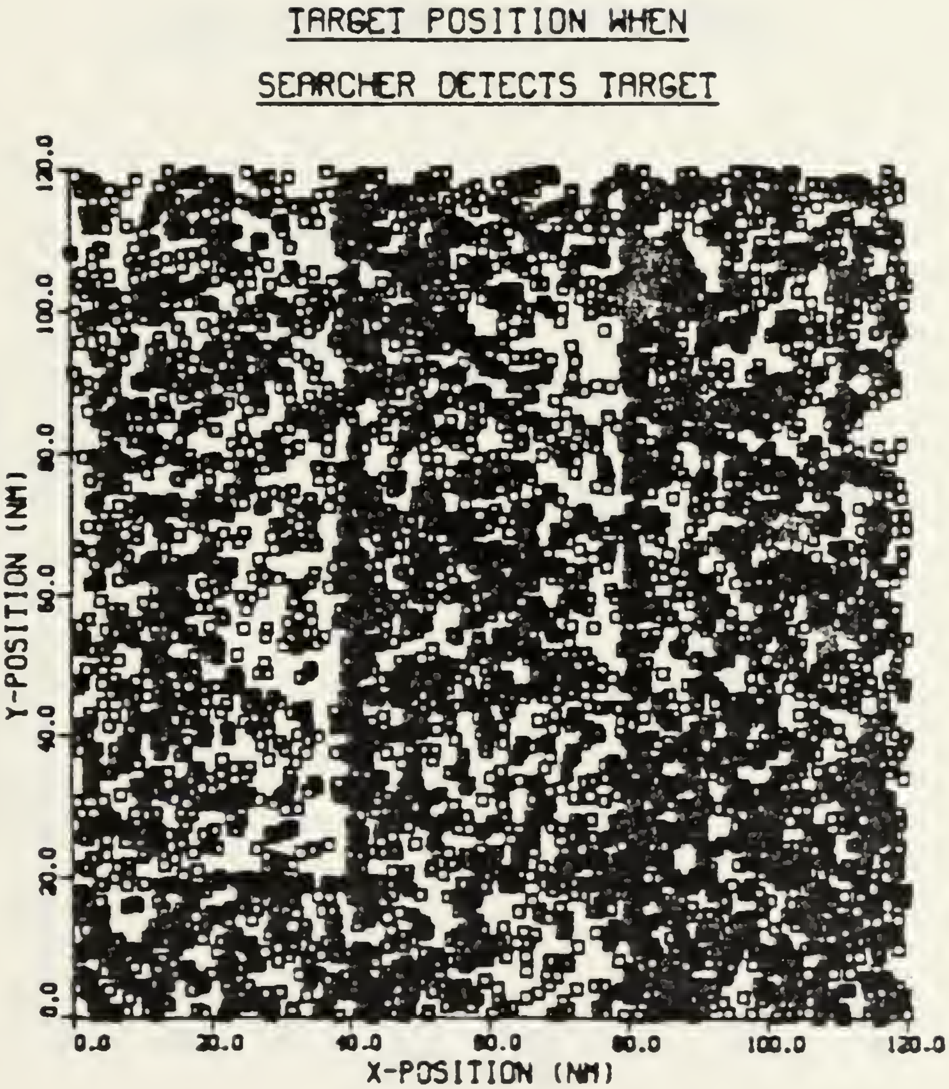

Figure 3.49. Target Position at Time of Detection for C-II.

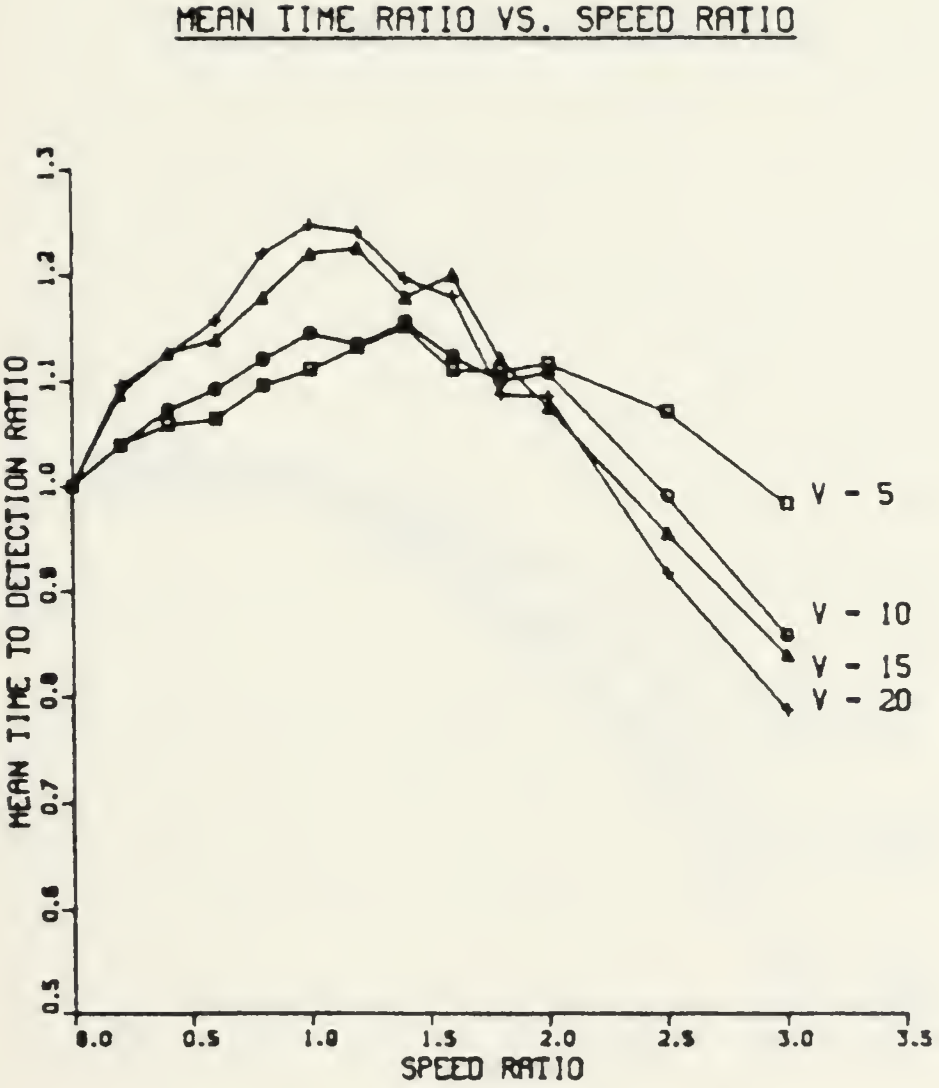

Figure 3.50. Mean Time Ratio Vs. Speed Ratio.

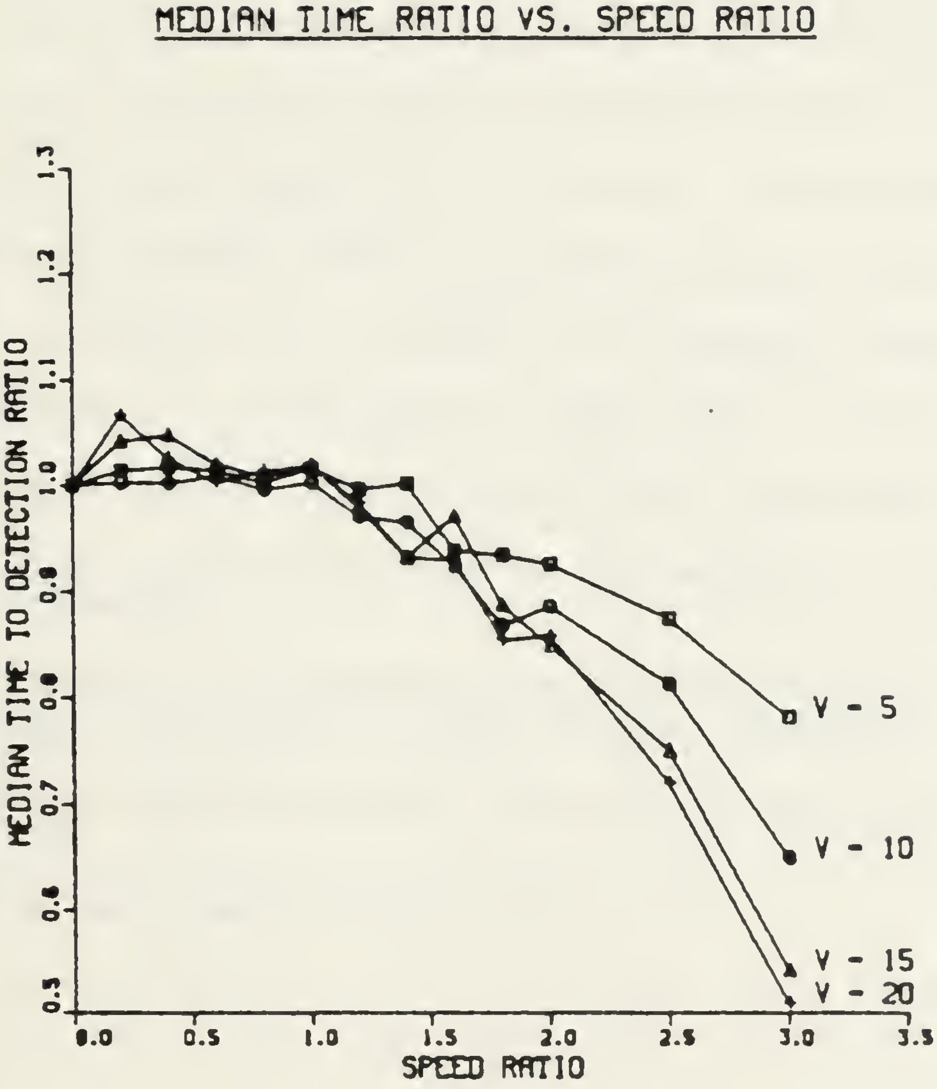

Figure 3.51. Median Time Ratio Vs. Speed Ratio.

E. CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

In the preceeding examples, it was intended to demonstrate the capabilities and flexibility of the PASS program. Of significance is the fact that an experienced operator, using the interactive data entry procedures, could run all of the example problems in about four hours (including time for hard-copy printing and graphics production). These runs represent 274,000 individual simulation replications.

The utility of the graphical output should be evident. Frequently the pictorial displays provide insight that is obscured by a large quantity of numbers. Additionally, the very shape of certain graphical results inspire questions the answer to which are critical to a thorough analysis.

Improvements or additions to PASS might include:

1. Multiple sensor capability by both platforms.
2. Multiple search platforms (e.g. searcher, target, and consort).
3. Post detection analysis (e.g. break contact, approach and attack, target motion analysis).
4. Gaming capability (real time control over platform motion).
5. Weapon employment capability.

APPENDIX A

PASS USERS GUIDE

A. INTRODUCTION

This User's Guide is a set of instructions designed to allow the execution of program PASS on the NPGS computer by users unfamiliar with the program code. The user should be familiar with the concepts of the simulation model as presented in Chapter I, and have available for reference the variable list in Appendix B.

1. Data File Management

When PASS is run, an input data file may, or may not exist. If no data file exists, the easiest way to create one is to execute program PASS and input data during the interactive session. If a data file exists it must have the filename PASS and filetype DATA. If a data file is interactively created, it will be in file PASS DATA at the completion of the run of PASS.

At the start of each run, if PASS DATA exists, it is copied into file PASS HOLD. If changes are made to the input data, file PASS DATA will reflect these changes, and PASS HOLD will contain the unchanged data. If no changes were made, PASS HOLD and PASS DATA will be identical. To save the data in PASS HOLD, it must be renamed before the next run of PASS. PASS DATA will remain unchanged as long as no changes are made to the data during the interactive session.

Upon completion of each PASS run a file PASS OUTPUT is created. Depending on the option selected by the user, this file will contain some, or all of the following:

1. A complete record of the input data, or a user specified run identification number.
2. A summary of the results of the simulation.
3. Sectioning results of times and ranges of detection and counter-detection (requires a minimum of 500 data points).
4. Histogram and statistics of times and ranges of detection and counter-detection (requires a minumum of 10 data points).

If this file is to be saved, it must be renamed or printed before the next run of PASS.

Upon completion of each run of PASS a file PASPLT DATA is created. This file provides the data, in the proper format, to be used by the FORTRAN program PASPLT. Program PASPLT uses the DISSPLA graphics system resident on the NPGS computer to produce a graphical display of the simulation input parameters and results.

2. Required Input Data

PASS requires the following data, which the user should have ready for interactive data input if a complete PASS DATA file does not exist:

1. Size of the search area rectangle.
2. Searcher track anchor points and return point.
3. Searcher propagation loss data (direct path and CZ's).
4. Target propagation loss data (direct path and CZ's).

5. Target initial x-distribution if not uniform (barrier scenario only).
6. Searcher figure-of-merit at sprint and drift speeds (as a function of target aspect, if target source level is aspect dependent).
7. Searcher sprint and drift speeds.
8. Searcher sprint and drift times.
9. Target figure-of-merit against sprinting and drifting searcher.
10. Target speed range.
11. Rate of target zigs (course and speed changes).
12. Acoustic fluctuation rate parameters.
13. Acoustic fluctuation scale parameters.
14. Run parameters (random number seed, number of replications, maximum search time for each replication).

During the interactive input of data, the program does some error checking. However, it is far from impossible to enter bad data, or incorrectly formatted data. If the program fails to run as expected, the most probable cause is bad data.

3. The Executive Program

The executive program PASS EXEC is necessary to run the FORTRAN program PASS. The executive program conducts data file management and starts PASS. Prior to invoking the executive program, PASS must be compiled in either FORTRAN-G or FORTRAN-H with the text file on the A-DISK. The FORTRAN-H compiler is recommended as it results in a faster program run time.

B. PROGRAM PASS EXECUTION

Each of the following sections describes a terminal option. Each option presents a new menu on the terminal screen. Depending on your response to each option menu, you will proceed to the next option, or you will be prompted to enter data or make some decision regarding the nature of the simulation.

1. Starting the Program: Invoking the Executive Program

To start program PASS, ENTER: PASS

The screen will contain a quick review of data-file management and the first OPTION.

2. OPTION NO. 1: Terminate or Proceed

This option allows you to "gracefully" terminate the program.

To terminate the program, ENTER: 1

To proceed with execution, ENTER: 2

If you terminate now, and you had an input data file PASS DATA, it will have been copied into file PASS HOLD, and PASS DATA will be empty.

3. MASTER OPTION: Accept Program Defaults

This option allows you to bypass the interactive sections and proceed directly to the simulation. Bypassing requires a complete and properly formatted input data file, and the acceptance of the program defaults.

To list the program defaults, ENTER: 1

To accept the program defaults, ENTER: 2

To commence interactive options, ENTER: 3

4. OPTION NO. 2A: Signal Integration Model

This option allows you to select a threshold crossing model without signal integration, or the MSEL integration model. It is recommended that the MSEL model be used if strong convergence zones are present.

To select no signal integration, ENTER: 1

To select the MSEL integration model, ENTER: 2

5. OPTION NO. 2: Acoustic Fluctuation Model

This option allows you to select either a Lambda-Sigma Jump or a Gauss-Markov error process to model acoustic fluctuations.

To select the Lambda-Sigma Jump model, ENTER: 1

To select the Gauss-Markov model, ENTER: 2

6. OPTION NO. 3A: Existence of File PASS DATA

The program must be "told" whether or not to read data from an existing data file.

If PASS DATA is not on your A-DISK, ENTER: 1

If PASS DATA is on your A-DISK, ENTER: 2

7. OPTION NO. 3: Search Area Size

Variables: XMAX, YMAX

This option allows you to specify the size of the search area in which the target is confined. The area dimensions are in nautical miles.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

8. OPTION NO. 4: Searcher Track

Variables: NP, KP, XP(I), YP(I)

This option allows you to enter the searcher track anchor points (2 to 50 points), and the return point. The dimensions of the anchor points are nautical miles from (0, 0).

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen

for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or futher modify data). To return, ENTER: 0.

9. OPTION NO. 5: Searcher Direct Path Propagation Loss

Variables: MO, RO(I), OL(I)

This option allows you to input the propagation loss of the searcher as a function of range. The number of points should be between 2 and 20, ensuring that there are sufficient points to adequately describe the propagation loss curve. Range is in nautical miles, and propagation loss in decibels (db).

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

10. OPTION NO. 6: Target Direct Path Propagation Loss

Variables: MT, RT(I), OL(I)

This option is similar to OPTION 5, but here you enter the propagation loss data for the target.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

11. OPTION NO. 7: Searcher Convergence Zones

Variables: NCZO, RCZO(I,J), CZLO(I)

This option allows you to input the number of searcher convergence zones (0 to 5), the width of each CZ, and the propagation loss in each CZ. The CZs are modeled as inverted square-wells superimposed on the DP propagation loss curve. You will have to make a subjective interpretation of the actual CZ data to determine the range to the inner and outer rings of the square-well, and the effective propagation loss therein. Ranges are in nautical miles, and propagation loss in db.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

12. OPTION NO. 8: Target Convergence Zones

Variables: NCZT, RCZT(I,J), CZLT(I)

This option is similar to OPTION 7, but here you enter the data for target convergence zones.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

13. OPTION NO. 9: Area or Barrier Search

This option allows you to select either an area or barrier search scenario. If area search is selected, the next option will be 10A. If barrier search is selected, the next option will be 9A.

To select area search scenario, ENTER: 1

To select barrier search scenario, ENTER: 2

14. OPTION NO. 9A: Initial Target Lateral Distribution

Variables: NBINS, XBIN(I), PBIN(I)

This option allows you to specify the lateral distribution of target starting position (barrier scenario only). Distances are in nautical miles from X = 0.

To distribute the initial positions uniformly on (0,XMAX), ENTER: 1.

To specify the lateral distribution, ENTER: 2

15. OPTION NO. 10A: Searcher FOM Dependent on Target Aspect

Variables: NLS, BRG(I), FOMBD(I), FOMBS(I)

This option allows you to specify the searcher figure-of-merit (at drift and sprint speeds) as a function of the relative bearing from target to searcher. This is equivalent to a target source level dependent upon aspect angle. Angles are in degrees, and FOM in db.

To specify searcher FOM not target aspect dependent, ENTER: 1

To specify searcher FOM target aspect dependent, ENTER: 2

If (1) is selected, OPTION 10 will be presented next. If (2) is selected, after data entry, OPTION 11 will be presented next.

16. OPTION NO. 10: Searcher Figure-of-Merit

Variables: FOMOS, FOMOD

This option allows you to specify the searcher figure-of-merit against the target when the searcher is at sprint or drift speeds. FOM IN db.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

17. OPTION NO. 11: Searcher Sprint and Drift Speeds

Variables: SOS, SOD

This option allows you to select the search speed (sprint and drift). For constant speed search, set SOS = SOD. Speeds in knots.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing menu will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

18. OPTION NO. 12: Searcher Sprint and Drift Times

Variables: TS, TD

This option allows you to specify the time the searcher spends at sprint and drift speeds. Time in hours.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

19. OPTION NO. 13: Target Figure-of-Merit

Variables: FOMTS, FOMTD

This option allows you to specify the target figure-of-merit against the searcher when the searcher is at sprint or drift speed. FOM in db.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option

menu (to review, accept, or further modify data). To return, ENTER: 0.

20. OPTION NO. 14: Target Speed Range

Variables: STMIN, STMAX

This option allows you to specify the range of target spped. Target speed is distributed uniform on (STMIN,STMAX). Speed in knots.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

21. OPTION NO. 15: Rate of Target Zigs

Variables: RTCC, RTSC

This option allows you to specify the rate of target course and speed changes. Rates are "per-hour".

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

22. OPTION NO. 16: Acoustic Fluctuation Rate Parameters

Variables: ALAM(I)

This option allows you to specify the rate parameters for the acoustic fluctuation process. Rates are "per-hour".

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen

for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

23. OPTION NO. 17: Acoustic Fluctuation Scale Parameters

Variables: SIGMA(I)

This option allows you to specify the scale parameters for the acoustic fluctuation process. Scale in db.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or futher modify data). To return, ENTER: 0.

24. OPTION NO. 18: Run parameters

Variables: SEED, NREP, TMAX

This option allows you to select parameters that govern the simulation execution and random number generation. Time in hours.

To review existing data, ENTER: 1

To accept existing data, ENTER: 2

To change existing data, ENTER: 3

If (1) is entered, the existing data (in file PASS DATA on your A-DISK) will be presented on the screen for review. To return to the option menu to accept or modify this data, ENTER: 0.

If (2) is entered, the existing data will be read into PASS, and the next option menu will be presented on the screen.

If (3) is entered, prompting messages will be on the terminal screen to allow you to enter new data. When data entry is completed, you must return to the option menu (to review, accept, or further modify data). To return, ENTER: 0.

25. OPTION NO. 19: Input Data to Output File

This option allows you to either send all of the input data to the output file, or to identify the output file with a run identification number.

To send all input data to the output file, ENTER: 1

To supress input data echo and select a run ID, ENTER: 2

26. OPTION NO. 20: Statistical Analysis

This option allows you to select the amount and type of statistical analysis to be done on the simulation results.

For complete statistical analysis (sectioning, histograms, and parameter point estimates, ENTER: 1

To omit sectioning, ENTER: 2

For no statistical analysis, ENTER: 3

C. NORMAL RUNNING INDICATION

Immediately following OPTION 20, the screen will display a running total of the number of replications completed. This display is updated every 200 replication, which allows the user to follow the progress of program execution. Upon completion of the last replication, the screen will be cleared, and the simulation quick-summary will be presented.

D. GRAPHICAL PRESENTATION OF RESULTS: PROGRAM PASPLT

The FORTRAN program PASPLT is intended to be used in conjunction with the DISSPLA graphics system to produce graphical representations of the results of a run of program PASS. To run program PASPLT you will need:

1. Data file PASPLT DATA on your A-DISK. PASPLT DATA is automatically created by running program PASS.
2. Dual terminal graphics capability (TEKTRONIX 618 and thermal printer).
3. Program PASPLT FORTRAN-G or FORTRAN-H compiled on your A-DISK.

To produce graphical output, invoke the graphics package by entering: DISSPLA. Two responses to the DISSPLA executive program prompts are required:

1. You must identify the program PASPLT as the FORTRAN program to be used with DISSPLA.
2. You must define the data file as follows: FILEDEF 10 DISK PASPLT DATA.

The above is done in response to prompting messages produced on the screen after DISSPLA is entered.

Each plot is presented on the TEKTRONIX 618 screen, and hard copies may be made by pressing the HARD COPY button at the bottom of the screen. To produce the next plot, hit ENTER on the keyboard. If any messages (other than error messages) appear on the IBM screen, clear the screen until the next plot appears on the TEKTRONIX screen.

APPENDIX B

DESCRIPTION OF VARIABLE NAMES USED IN PASS

The following list of major variable names used in PASS is arranged alphabetically. Those variable names with an asterik (*) indicate that an understanding of their function is required to run PASS in conjunction with Appendix A (PASS User's Guide). Following each variable name is the data type and physical dimension, if any, and then a verbal description of the variable use. For input data, restrictions (e.g. variable size, entry procedures) are listed under "RESTR:". Some variables used for temporary storage, loop counters, or pointers are not listed as their function should be apparent to the reader of the code.

PASS VARIABLE NAMES:

1. AFL(I)/real/decibels/.....Value of the Ith acoustic fluctuation process. I=1,2,3.
2. AFLGM/logical/.....If the Gauss-Markov error process is to be used , this variable is TRUE. Otherwise, it is false.
3. AFLLS/logical/.....If the Lambda-Sigma Jump error process is used, this variable is TRUE. Otherwise, it is FALSE.
4. ALAM(I)/*/real/inverse hours/.....Rate parameter for the Ith fluctuation process, I=1,2,3, (input). RESTR: positive.
5. ALFA/real/.....Multiplier for the GMA process.

6. ANG1/real/degrees(external), radians(internal)/.....A limit of target courses allowed in the barrier scenario. In the simulation, courses are measured CCW from the X-axis. In the barrier scenario, the target base course is in the negative Y-direction, or psuedo-course of 270-degrees. ANG1 = 270 - ANG2.

7. ANG2/*/real/degrees(external), radians(internal)/..... Angular variation from base course (ANG1) allowed for the target in the barrier scenario. The target courses are distributed uniformly over psuedo-course 270±ANG2. See ANG1 (input). RESTR: non-negative, not greater than 90-degrees.

8. ASPECT/logical/.....If the target source level is aspect dependent, this variable is TRUE. Otherwise, it is FALSE.

9. BETA/real/.....Multiplier for the GMA process.

10. BREL/real/radians/.....Relative bearing from the target to the searcher.

11. BRG(I)/*/real/degrees(external), radians(internal)/ .....Relative bearing associated with target aspect related searcher figure-of-merit, I = 1,...,NLS, (input). RESTR: BRG(1) must be 000.0 (directly on the bow), and all subsequent BRG(I) must be in ascending order, such that BRG(I) < BRG(I+1), and BRG(I) max = 360.0.

12. C1,C2,C3,C4/integer/.....Counters for STACK1,2,3,4.

13. COSX/real/.....X-component of a unit vector in the direction of target motion.

14. CZLO(I)/*/real/decibels/.....Propagation loss in the Ith searcher convergence zone (input). If the number of searcher convergence zones (NCZO) is zero, then this data is not input. Otherwise, I=1,...,NCZO. RESTR: positive.

15. CZLT(I)/*/real/decibels/.....Propagation loss in the Ith target convergence zone (input). If the number of target convergence zones (NCZT) is zero, then this data in not input. Otherwise, I=1,...NCZT. RESTR: positive.

16. D1/real/nautical miles/.....Distance from current searcher position (XO,YO) to the projected searcher position (XOT,YOT).

17. D2/real/nautical miles/.....Distance from current searcher position (XO,YO) to the end of the current search leg.

18. DIST(I)/real/nautical miles/.....The distance from searcher track anchor point I to anchor point (I+1). The length of the Ith leg of the search pattern.

19. DX(I)/real/.....X-component of the unit vector in the direction of searcher motion from anchor point I to anchor point (I+1).

20. DY(I)/real/.....Y-component of the unit vector pointing in the direction of searcher motion from anchor point I to anchor point (I+1).

21. ECHOS/logical/.....If the user chooses to have the input data sent to the output file, this variable is TRUE. Otherwise, it is FALSE.

22. ENUF/logical/.....If 5 hours of figure-of-merit data for the target has been recorded, this variable is TRUE. Otherwise, it is FALSE.

23. ENUFF/logical/.....If 5 hours of figure-of-merit data for the searcher has been recorded, this variable is TRUE. Otherwise, it is FALSE.

24. FO(I)/real/decibels/.....A sample of the searcher figure-of-merit for the first five hours of a replication used to provide a representative plot of FOM as a reference. This variable is not used computationally in the program.

25. FOMBD(I)/*/real/decibels/.....Searcher figure-of-merit when at drift speed and with a relative bearing from the target to the searcher of BRG(I), where I = 1,...,NLS, (input). RESTR: positive.

26. FOMBS(I)/*/real/decibels/.....Searcher figure-of-merit at sprint speed at the corresponding relative bearing from target to searcher of BRG(I), I = 1,..., NLS, (input). RESTR: positive.

27. FOMO/real/decibels/.....Current searcher figure-of-merit.

28. FOMOD/*/real/decibels/.....The searcher figure-of-merit at drift speed (input). RESTR: positive.

29. FOMOS/*/real/decibels/.....The searcher figure-of-merit at sprint speed (input). RESTR: positive.

30. FOMT/real/decibels/.....Current target figure-of-merit.

31. FOMTD/*/real/decibels/.....The target figure-of-merit when the searcher is at drift speed (input). RESTR: positive.

32. FOMTS/*/real/decibels/.....The target figure-of-merit when searcher is at sprint speed (input). RESTR: positive.

33. FILE/logical/.....This variable is TRUE if the user specifies a data file, PASS DATA, presently exists on his A-disk. Otherwise, it is FALSE.

34. FT(I)/real/decibels/.....A sample of the target figure-of-merit for the first five hours of a replication used to provide a plot of representative FOM as a reference. This variable is not used computationally elsewhere in the program.

35. GO4IT/logical/.....This variable is TRUE if the user specifies that the interactive data input is to be bypassed and input data is to be read directly from an existing file on A-disk, PASS DATA. Otherwise, it is FALSE.

36. ICZO/integer/.....Identifies the number of the convergence zone in which the searcher CZ detection took place.

37. ICZT/integer/.....Identifies the number of the convergence zone in which the target CZ counter-detection took place.

38. INTEG/logical/.....This variable is TRUE if the MSEL model is to be used. Otherwise, it is FALSE.

39. KBAR/integer/.....Indicator variable. KBAR = 1 if the area search scenario is used. KBAR = 2 if the barrier search scenario is used.

40. KCZO/integer/.....The number of the most distant CZ in which the searcher can currently make a detection. If KCZO = 0, no CZ detections are currently possible. If the searcher makes a CZ detection, KCZO = -1.

41. KCZT/integer/.....The number of the most distant CZ in which the target can currently make a detection. If KCZT = 0, no CZ detections are currently possible. If the target makes a CZ detection, KCZT = -1.

42. KP/*/integer/.....The number of the searcher track anchor point at which the search pattern begins to repeat. This is the "return point" from which subsequent patterns will start after the first full pattern is executed (input). RESTR: positive, less than NP. See NEXT(I) and NP.

43. MO/*/integer/.....The number of range/propagation loss data points used to define the searcher direct-path (DP) propagation loss curve (input). RESTR: greater than 1, and less than 21.

44. MODE/integer/.....Indicator variable. If MODE = 1, searcher is at drift speed. If MODE = 2, searcher is at sprint speed.

45. MSELO/integer/.....The MSEL counter for the searcher.

46. MSELT/integer/.....The MSEL counter for the target.

47. MT/*/integer/.....The number of range/propagation loss data points used to define the target direct path-propagation loss curve (input). RESTR: greater than 1, and less than 21.

48. NBINS/*/integer/.....The number of sections, each with a fixed probability, that define the initial target distribution across the top of a barrier choke point (input). RESTR: greater than 1, and less than 21.

49. NBOTH/integer/.....The number of simultaneous detections.

50. NCZDO(I)/integer/.....In the main program, the number of searcher detections made in the Ith convergence zone. Converted in subroutine SINKEM to the fraction of searcher detections in the Ith CZ.

51. NCZDT(I)/integer/.....In the main program, the number of target counter-detections made in the Ith convergence zone. Converted in subroutine SINKEM to the fraction of target detections in the Ith CZ.

52. NCZO/*/integer/.....The number of searcher convergence zones (input). RESTR: non-negative, less than 6.

53. NCZT/*/integer/.....The number of target convergence zones (input). RESTR: non-negative, less than 6.

54. NDO/integer/.....The number of detections made by searcher.

55. NDT/integer/.....The number of detections made by the target.

56. NEXT(I)/integer/.....The next anchor point in the searcher track after point I. The program sets NEXT(I) = I+1 for I = 1 to NP-1, and NEXT(NP) = KP. Thus, after one complete execution of the search pattern, the searcher transits from point NP to KP, and repeats the pattern starting at KP.

57. NL/integer/.....Pointer to the next searcher track leg (similar to NLEG).

58. NLEG/integer/.....A pointer to indicate the searcher position in the search pattern. The searcher is always between anchor point NLEG and point NEXT(NLEG).

59. NLS/*/integer/.....The number of relative bearing/figure-of-merit data points used to define the radial distribution of target source level resulting in a radial distribution of searcher figure-of-merit (input). RESTR: greater than 2, less than 51.

60. NONE/integer/.....The number of replications in which neither ship makes a detection prior to TMAX.

61. NP/*/integer/.....The number of searcher track anchor points (input). RESTR: greater than 1, less than 51.

62. NREP/*/interger/.....The number of replications to be run (input). RESTR: positive, less than 5001.

63. NTDO(I)/real/hours/.....The time of the Ith searcher detection.

64. NTDT(I)/real/hours/.....The time of the Ith target counter-detection.

65. OFOM(I)/real/decibels/.....Through the EQUIVALENCE statement, OFOM(1) = FOMOD, OFOM(2) = FOMOS. The subscript of this variable in the program is MODE. See MODE, FOMOD, FOMOS.

66. OL(I)/*/real/decibels/.....Direct-path propagation loss at range RO(I) for the searcher (input). RESTR: OL(I) must increase as range increases. OL(I) < OL(I+1).

67. PBIN(I)/*/real/.....The probability of target inclusion in the Ith section when the user defines initial lateral target distribution, I = 1,...,NBINS, (input). RESTR: non-negative, less than or equal to 1.0. Sum of all PBIN(I) = 1.0.

68. RCZO(I,J)/*/real/nautical miles/.....Range to the inner (J=1) and outer (J=2) edges of the Ith convergence zone for the searcher, I=1,...,NCZO, (input). RESTR: RCZO(I,1) < RCZO(I,2) < RCZO(I+1,1) < RCZO(I+1,2), etc.

69. RCZT(I,J)/*/real/nautical miles/.....Range to the inner (J=1) and outer (J=2) edges of the Ith convergence zone for the target, I = 1,...,NCZT, (input). RESTR: RCZT(I,1) < RCZT(I,2) < RCZT(I+1,1) < RCZT(I+1,2), etc.

70. RFLCT/logical/.....RFLCT is FALSE unless a target reflection takes place.

71. RMAX/real/nautical miles/.....The maximum of RNGO,RNGT.

72. RNG/real/nautical miles/.....Current range from searcher to target.

73. RNGO/real/nautical miles/.....Searcher current direct path detection range.

74. RNGT/real/nautical miles/.....Target current direct path detection range.

75. RNTDO(I)/real/nautical miles/.....Range between searcher and target when searcher detects target.

76. RNTDT(I)/real/nautical miles/.....Range between searcher and target when target detects searcher.

77. RO(I)/*/real/nautical miles/.....Range associated with searcher propagation loss OL(I) (input). RESTR: Must be entered in ascending order, RO(I) < RO(I+1).

78. RT(I)/*/real/nautical miles/.....Range associated with target propagation loss TL(I) (input). RESTR: Must be entered in ascending order, RT(I) < RT(I+1).

79. RTCC/*/real/inverse hours/.....Rate of target course changes. 1/RTCC is the mean of an exponential distribution from which the time interval to target course change is drawn (input). RESTR: positive.

80. RTSC/*/real/inverse hours/.....Rate of target speed changes. 1/RTSC is the mean of an exponential distribution from which the time interval to target speed change is drawn (input). RESTR: positive.

81. SEED/*/integer/.....Psuedo-random number generator seed (input). RESTR: non-negative.

82. SIGMA(I)/*/real/decibels/.....Scale parameter for the Ith acoustic fluctuation process, I=1,2,3. RESTR: non-negative.

83. SINX/real/.....The y-component of a unit vector in the direction of target motion.

84. SO/real/knots/.....Current searcher speed.

85. SOD/*/real/knots/.....Searcher drift speed (input). For constant speed search, set drift speed equal to sprint speed (SOD = SOS).

86. SOS/*/real/knots/.....Searcher sprint speed (input). For constant speed search, set sprint speed equal to drift speed (SOS = SOD).

87. SPECX/logical/.....This variable is TRUE if the user has specified a lateral target distribution other than uniform across the barrier (barrier scenario only). Otherwise, it is FALSE.

88. SPEED(I)/real/knots/.....Through the EQUIVALENCE statement, SPEED(1) = SOD, SPEED(2) = SOS. The subscript of this variable in the program is MODE. See MODE, SOS, SOD.

89. ST/real/knots/.....Target current speed, drawn from a uniform distribution on the interval (STMIN,STMAX).

90. STACK1,STACK2,STACK3,STACK4/real/.....Each is a stack of 5000 psuedo-random numbers which are sequentially dereferenced for program use by the stack counters C1,C2,C3,C4. STACK1 is distributed exponentially, mean = 1.0. STACK2 is sistributed normally, mean = 0.0, variance = 1.0. STACK3 is distributed uniformally on the interval (0,1). STACK4 is distributed as specified by the user.

91. START/*/real/hours/.....The latest time-late on barrier time. The time-late on barrier for the searcher is distributed uniformly on the interval (0,START), with an assumption that the target travels at mean speed from time zero until the searcher is on the barrier (input). RESTR: non-negative.

92. STATS1/logical/.....This variable is FALSE if no statistical analysis is to be done. Otherwise it is TRUE.

93. STATS2/logical/.....This variable is TRUE if sectioning of the data is to be accomplished. Otherwise, it is FALSE.

94. STINC/real/knots/.....The interval of speed which defines the uniform distribution of allowable target speeds (STMAX - STMIN).

95. STMAX/*/real/knots/.....Target maximum speed (input). RESTR: STMAX must be greater than or equal to STMIN.

96. STMIN/*/real/knots/.....Target minimum speed (input). RESTR: STMIN must be less than or equal to STMAX.

97. TBIG/real/nautical miles or hours/.....TBIG is used to compute the simulation time step. Initially, TBIG is a range = RNG - RMAX, and is positive if, and only if, no direct path detection is possible. If no detection is possible (CZ or DP), TBIG is set to the minimum closing distance such that either a detection or counter-detection is just possible (DP or CZ) based on the current acoustic conditions. TBIG is converted to a time interval by dividing by the current combined speeds of the searcher and target, which assumes a worst case head-to-head closing situation. The simulation time-step is then set to the maximum of TBIG or 0.05 hour.

98. TCC/real/hours/.....Time of next searcher course change.

99. TD/*/real/hours/.....Searcher drift time (input). RESTR: positive.

100. TFOM(I)/real/decibels/.....Through the EQUIVALENCE statement, TFOM(1) = FOMTD, TFOM(2) = FOMTS. The subscript of this variable in the program is MODE. See MODE, FOMTD, FOMTS.

101. THETA/real/radians/.....Target psuedo-course. THETA is measured from the x-axis in a counter-clockwise direction.

102. TIFL(I)/real/hours/.....For I=1,2,3, TIFL(I) is the time at which AFL(I) changes in the LSJ process, or the next sample time for the GMA process. For I = 4-7, through the EQUIVALENCE statement, TIFL(4) = TCC, TIFL(5) = TTCC, TIFL(6) = TTSC, TIFL(7) = TSC.

103. TIME(I)/real/hours/.....Through the EQUIVALENCE statement, TIME(1) = TD, TIME(2) = TS. The subscript of this variable in the program is MODE. See MODE, TD, TS.

104. TINC/real/hours/.....The time at which the next simulation time-step ends.

105. TL(I)/*/real/decibels/.....Propagation loss at range RT(I) for the target (input). TL(I) must increase with increasing range. TL(I) must be less than TL(I + 1).

106. TLAST/real/hours/.....The time at which the Gauss-Markov error process was last evaluated.

107. TMAX/*/real/hours/.....Maximum allowed search time per replication.

108. TNOW/real/hours/.....Current time in a replication.

109. TO(I)/real/hours/.....The time at which the figure-of-merit sample for the searcher, FO(I), was taken.

110. TOXMAX/real/nautical miles/.....Twice the value of XMAX.

111. TOYMAX/real/nautical miles/.....Twice the value of YMAX.

112. TS/*/real/hours/.....Searcher sprint time (input). RESTR: positive.

113. TSC/real/hours/.....Time of next searcher speed change.

114. TT(I)/real/hours/.....The time at which the target figure-of-merit sample, FT(I), was taken.

115. TTCC/real/hours/.....Time of next target course change.

116. TTSC/real/hours/.....Time of next target speed change.

117. TWOPI/real/.....Twice the value of π.

118. UX/real/knots/.....The x-component of current target velocity vector.

119. UY/real/knots/.....The y-component of current target velocity vector.

120. VX/real/knots/.....The x-component of current searcher velocity vector.

121. VY/real/knots/.....The y-component of current searcher velocity vector.

122. XBIN(I)/*/real/nautical miles/.....The distance from the origin ($x = 0.0$) to the right-most x-value of the section containing target probability mass PBIN(I). RESTR: positive, XBIN(NBINS) = XMAX.

123. XMAX/*/real/nautical miles/.....Length of the search area rectangle in the x-direction (input). RESTR: positive.

124. XO/real/nautical miles/.....The current searcher x-position.

125. XODT(I)/real/nautical miles/.....The x-position of the searcher when the searcher makes the Ith detection of the target.

126. XOT/real/nautical miles/.....The projected searcher position based on the current position (XO) and the computed simulation time-step.

127. XOTD(I)/real/nautical miles/.....The x-position of the searcher when the target makes the Ith counter-detection of the searcher.

128. XP(I)/*/real/nautical miles/.....The x-position of the Ith searcher track anchor point (input).
RESTR: non-negative.

129. XT/real/nautical miles/.....The current x-position of the target.

130. XTDO(I)/real/nautical miles/.....The x-position of the target when the target makes the Ith counter-detection of the searcher.

131. XTOD(I)/real/nautical miles/.....The x-position of the target when the searcher makes the Ith detection of the target.

132. YMAX/*/real/nautical miles/.....Length of the search area in the y-direction (input).
RESTR: positive.

133. YO/real/nautical miles/.....Current searcher y-position.

134. YODT(I)/real/nautical miles/.....The y-position of the searcher when the searcher makes the Ith detection of the target.

135. YOT/real/nautical miles/.....Projected searcher y-position based on the current position (YO) and the simulation time-step.

136. YOTD(I)/real/nautical miles/.....The y-position of the searcher when the target makes the Ith counter-detection of the searcher.

137. YP(I)/*/real/nautical miles/.....The y-position of the Ith searcher track anchor point (input).
RESTR: positive.

138. YT/real/nautical miles/.....Current y-position of the target.

139. YTDO(I)/real/nautical miles/.....The y-position of the target when the target makes the Ith counter-detection of the searcher.

140. YTDO(I)/real/nautical miles/.....The y-position of the target when the searcher makes the Ith detection of the target.

APPENDIX C

ABBREVIATIONS AND ACRONYMS

This appendix contains an alphebetical listing of abbreviations and acronyms used in this thesis.

1. ASW: Antisubmarine Warfare.
2. Cov(X,Y): If X,Y are random variables, this is the covariance function. If X,Y are realizations of an autoregressive time series, this is the autocovariance function.
3. CPU: Central Processing Unit.
4. CZ: Convergence Zone. Usually used to describe the propagation mode of an acoustic signal.
5. db: Decibel.
6. DP: Direct Path. Usually used to describe the propagation mode of an acoustic signal.
7. E(X): The expected value of the random variable X.
8. ER: Exchange Ratio.
9. EXP(x): An exponential distribution with rate parameter x.
10. FOM: Figure-of-Merit.
11. IMSL: International Mathematics and Statistics Library.
12. Le: Environmental noise.
13. Ls: Target noise source level.
14. MOEs: Measure Of Effectiveness for the Searcher
15. MOEt: Measure Of Effectiveness for the Target.
16. MSEL: Minimum Signal Excess Logic.

17. N(x,y): A normal distribution with mean x and variance y.

18. NONIMSL: Non-International Mathematics and Statistics Library.

19. NPGS(NPS): Naval Postgraduate School.

20. Nrd: Recognition differential.

21. PASS: Passive Acoustic Search Simulation.

22. PCD: Probability of Counter-Detection.

23. PCDCZ: Probability of Counter-Detection in a Convergence Zone.

24. PCDCZI: Probability of Counter-Detection in the Ith Convergence Zone.

25. PCDDP: Probability of Counter-Detection by Direct Path propagation.

26. PD: Probability of Detection.

27. PDCZ: Probability of Detection in a Convergence Zone.

28. PDCZI: Probability of Detection in the Ith Convergence Zone.

29. PDDP: Probability of Detection by Direct Path propagation.

30. $\rho(X,Y)$: If X,Y are random variables, ρ is the linear correlation coefficient. If X,Y are realizations of an autoregressive time series, then ρ is the autocorrelation function.

31. Rs/Rt: The mean detection/counter-detection range.

32. (R50)s/(R50)t: The median detection/counter-detection range.

33. SE: Signal Excess.

34. Ts/Tt: Mean time to detection/counter-detection.

35. (T50)s/(T50)t: Median time to detection/counter-detection.

36. TMA: Target Motion Analysis.

37. U(a,b): A uniform distribution on the interval (a,b).

38. Var(X): The variance of the random variable X.

APPENDIX D

APPLICATION OF THE PASSIVE SONAR EQUATION TO THE PASS MODEL

The passive sonar equation may be written as:

$$SNR = Ls - Nw - Le$$

where:

1. SNR = the signal-to-noice ration at the processor output (db).
2. Ls = the target source level (db).
3. Nw = the signal propagation loss (db).
4. Le = the background noise compensated for by sonar directivity (db).

Now, make the following definitions:

1. Nrd = RECOGNITION DIFFERENTIAL (db). Nrd is that SNR required for a probability of detection of 0.5. It can be thought of as the SNR required for a target to be recognized as a valid contact 50% of the time. Nrd is generally a function of sonar processor performance.
2. FOM = FIGURE-OF-MERIT (db). FOM is that amount of signal propagation loss (Nw) that results in a probability of detection of 0.5.
3. SE = SIGNAL EXCESS (db). SE is the algebraic difference of FOM and Nw (SE = FOM - Nw).

If we now substitute Nrd for SNR and FOM for Nw in the above equation, and solve for FOM, we have:

$$FOM = Ls - Le - Nrd$$

Thus, the result is:

$$SE = 0 \Leftrightarrow Nw = FOM \Leftrightarrow Pd = 0.5$$
$$SE > 0 \Leftrightarrow Nw < FOM \Leftrightarrow Pd > 0.5$$
$$SE < 0 \Leftrightarrow Nw > FOM \Leftrightarrow Pd < 0.5$$

In PASS, we obtain a certain detection whenever the sum of the deterministic signal excess and the error term is greater than or equal to zero. The error term has a zero mean, and the probability that it is greater than zero is 0.5. Thus, when the deterministic signal excess is zero, the probability of detection in PASS is 0.5, which is consistent with the passive sonar equation results above.

APPENDIX E

COMPUTER SIMULATION OF THE STOCHASTIC ERROR PROCESSES

In this appendix we examine the modeling of the acoustic fluctuation process independent of all other events in the simulation. In effect, we will ignore all events which affect the simulation time-step except for acoustic fluctuations.

A. THE LAMBDA-SIGMA JUMP PROCESS

In PASS there are three fluctuation processes simultaneously in progess.

$X(t)$ = searcher local error process.

$Y(t)$ = global error process

$Z(t)$ = target local error process

Each process is defined by a rate parameter $(\lambda_1, \lambda_2, \lambda_3)$ and a scale parameter $(\sigma_1, \sigma_2, \sigma_3)$. For illustrative purposes, assume we initially start the process at time = T with independent draws from normal distributions parameterized by the scale parameters. That is:

$X(T) = \eta_x$ where $\eta_x \sim N\ (o,\sigma_1^2)$

$Y(t) = \eta_y$ where $\eta_y \sim N\ (o,\sigma_2^2)$

$Z(t) = \eta_z$ where $\eta_z \sim N\ (o,\sigma_3^2)$

Also, at time = T, we determine the time interval to the next fluctuation level change, s, by independent draws from exponential distributions parameterized by the rate parameters. That is:

$$S_x = \tau_x \text{ where } \tau_x \sim \text{EXP}(\lambda_1)$$

$$S_y = \tau_y \text{ where } \tau_y \sim \text{EXP}(\lambda_2)$$

$$S_z = \tau_z \text{ where } \tau_z \sim \text{EXP}(\lambda_3)$$

The actual time of the fluctuation changes would then be:

$$t_x = S_x + T$$

$$t_y = S_y + T$$

$$t_z = S_z + T$$

If no detection takes place at time = T, the next time a detection can take place is the time of the first fluctuation change. Therefore, the simulation time-step , Δt, would be:

$$\Delta t = \min \{S_x, S_y, S_z\}$$

All simulation events (e.g. platform motion) would then be executed with time-step Δt, and the simulation time would be incremented by Δt.

The fluctuation level associated with Δt is then changed. Suppose, for example, that $t_x < t_y < t_z$. Then, we

would change the X fluctuation level by replacing the previous level with a draw from $N(0,\sigma_1^2)$, and obtain a new time interval to the next change of X by making a draw from $EXP(\lambda_1)$, from which the time of the next change in X is calculated. If no detection takes place, a new time-step is computed as before, and the process is repeated.

B. THE GAUSS-MARKOV PROCESS

Unlike the LSJ process, the GMA process involves continuous sample paths. The defining function of the GMA process is of the form:

$$X(t + s) = e^{-\lambda s} X(t) + g(s)\eta_x$$

where:

1. η_x is a normally distributed random variable with zero mean and variance σ_1^2.
2. $g(s)$ is a function of the time increment, s, which has the properties:

$$\lim_{s \to o} g(s) = 0 \quad \text{and} \quad \lim_{s \to \infty} g(s) = 1$$

we know that:

$$\mathrm{Var}[X(t)] = \sigma_1^2 \ \forall t$$

so

$$\mathrm{Var}[X(t+s)] = \sigma_1^2 = \mathrm{Var}\ [e^{-\lambda s} X(t) + g(s)\eta_x]$$

Since $e^{-\lambda s}$ and g(s) are not random variables and X(t) and η_x are independent, we can write:

$$\sigma_1^2 = E^{-2\lambda s}\ \mathrm{Var}[X(t)] + [g(s)]^2\ \mathrm{Var}[\eta_x]$$

$$\sigma_1^2 = e^{-2\lambda s}\ \sigma_1^2 + [g(s)]^2\ \sigma_1^2$$

$$g(s) = (1-e^{-2\lambda s})^{1/2}$$

Thus, the functional form of GMA is:

$$X(t+s) = \rho X(t) + (1-\rho^2)^{1/2}\ \eta_x$$

where:

$$\rho = e^{-\lambda s}$$

The difficulty in using this process in a computer simulation is deciding on the time-step. The more accurately one wants to model the process, the smaller the time-step required, but small time-steps cause excessively long run times. As a practical compromise, the PASS model uses the identical method for determining fluctuation time-steps in both the LSJ and GMA process (i.e. the GMA process is sampled at exponentially distributed random times governed by the rate parameters). The difference in applying this to the GMA process is that each time one fluctuation process is changed in LSJ, all three are updated in GMA.

For example, in the previous section where X was changed, the GMA process would update X, Y, and Z based on a time increment s_x, and independent draws to η_x, η_y, η_z.

C. GENERATION OF PSUEDO·RANDOM NUMBERS

Psuedo random numbers used throughout PASS are generated using the NPGS random number package LLRANDOMII. This package was used, vice a portable package (e.g. International Mathematics and Statistics Library) because of the improvement in speed in number generation. The IBM Assembly Language LLRANDOMII routines are between three and eight times as fast as the FORTRAN IMSL subroutines.

D. CORRELATION OF THE ERROR PROCESSES

Suppose we define two compound error processes, A and B, as follows:

$$A(t) = X(t) + Y(t)$$

$$B(t) = Y(t) + Z(t)$$

where X, Y, Z are LSJ or GMA processes. Dropping the "t" notation for convenience, a computational formula for the covariance is:

$$\mathrm{Cov}(A,B) = E[A\cdot B]-E[A]\cdot E[B]$$

and since $E[A] = E[B] = 0$

$$\mathrm{Cov}(A,B) = E[A\cdot B] = E[XY+Y^2+XZ+YZ]$$

$$\mathrm{Cov}(A,B) = \mathrm{Cov}(X,Y) + \sigma_y^2 + \mathrm{Cov}(X,Z) + \mathrm{Cov}(Y,Z)$$

$$\mathrm{Cov}(A,B) = \sigma_y^2$$

since X,Y,Z are independent, and $E[Y] = 0$.

The linear coefficient of correlation is defined as:

$$\rho(A,B) = \frac{\mathrm{Cov}(A,B)}{\sigma_A \sigma_B}$$

where

$$\sigma_A = (\sigma_1^2 + \sigma_2^2)^{1/2}$$

$$\sigma_B = (\sigma_2^2 + \sigma_3^2)^{1/2}$$

so, the coefficient of correlation for the compound error process is:

$$\rho(A,B) = \left\{ \left(\frac{\sigma_1^2}{\sigma_2^2} + 1 \right) \left(\frac{\sigma_3^2}{\sigma_2^2} + 1 \right) \right\}^{-1/2}$$

Note that: 1. $\lim\limits_{\frac{\sigma_1}{\sigma_2}, \frac{\sigma_3}{\sigma_2} \to 0} \rho(A,B) = 1$

which satisfies the intuitive and practical requirement that as the global fluctuation becomes dominant, the error signal experienced by both sensors will be highly correlated.

2. $\lim\limits_{\frac{\sigma_1}{\sigma_2}, \frac{\sigma_2}{\sigma_2} \to \infty} \rho(A,B) = 0$

which satisfies the intuitive and practical requirement that as the local fluctuations dominate, the error signal experienced by the sensors will be almost uncorrelated.

APPENDIX F

A THREE-OUT-OF-FIVE DETECTION CRITERIA MODEL

The Minimum Signal Excess Logic (MSEL) starts in state zero (MSEL 0). Each time the acoustic environment is samples, the MSEL state is incremented by one if detection is possible (SE > 0). If detection is not possible (SE < 0), then the MSEL counter is decremented by one, unless MSEL is in state zero. The lowest MSEL state is zero, and detection occurs the first time the MSEL state reaches three. This logic results in a detection occuring if, and only if, three of the last five samples of the acoustic environment had SE ≥ 0.

MSEL is equivalent to the Markov chain shown in Figure F.1, where the transition probabilities are, in general, a function of time (i.e. they will change each time the acoustic signal is sampled). In using the LSJ process, this presents no problem in that the fluctuation process is static between samples. In using the GMA process, however, the fluctuation process is dynamic between sample times, and therefore the rate of sampling will have an effect on the time of detection. The PASS model accounts for this in two ways:

1. The rate of environment sampling is the same for LSJ and GMA. See Appendix E.
2. Whenever either searcher or target are not in MSEL 0, the maximum PASS time-step defaults to 0.05 hour (3 minutes).

For analysis purposes, assume all the transition probabilities are constant and equal to p. Let q = (1-p) and let Si = the mean number of time increments (samples of the acoustic environment) to go from state i to state 3. Reaching the absorbing state, state 3, is equivalent to detection.

We can write a series of balance equations as:

$$S_2 = p(1) + q(1 + S_1) = 1 + qS_1$$

$$S_1 = p(1 + S_2) + q(1 + So) = 1 + pS_2 + qSo$$

$$So = p(1 + S_1) + q(1 + So) = 1 + pS_1 + qSo$$

If we solve this system of equations for So, we have:

$$So = \frac{1}{p}(2 + \frac{1}{p^2})$$

Since So is the mean number of time steps to go from MSEL state zero to MSEL state 3 (for a fixed p), we can place an upper bound on the expected time to go from MSEL 0 to MSEL 3 as:

$$To = 3So \text{ (in minutes)}$$

since the maximum time-step is PASS is three minutes when the searcher or target MSEL counter is greater than zero. Table VIII shows representative values of So and To as a function of the instantaneous probability of detection. Keeping in mind the simplifying assumptions made, the values of To are analogous to maximum expected integration times.

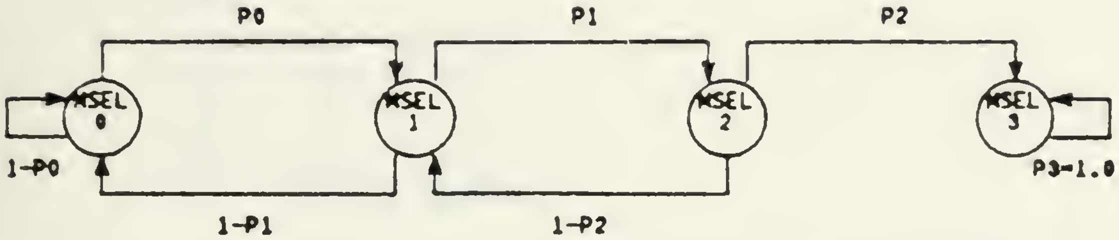

Figure F.1. The MSEL Model as a Markov Chain.

TABLE VIII

MSEL MAXIMUM EXPECTED INTEGRATION TIMES

p	So	To
1.0	3.000	9.00
0.9	3.594	10.78
0.8	4.453	13.36
0.7	5.773	17.32
0.6	7.963	23.89
0.5	12.000	36.00
0.4	20.625	61.88
0.3	43.704	131.11
0.2	135.000	405.00
0.1	1020.000	3060.00
0.0	∞	∞

APPENDIX G

CODE FOR EXECUTIVE PROGRAM TO RUN PASS

```
&TRACE
CP TERMINAL LINESIZE 80
ERASE PASS OUTPUT
ERASE PASS HOLD
ERASE PASPLT DATA
COPYFILE PASS DATA A1 = HOLD =
ERASE PASS DATA
FILEDEF 06 DISK PASS OUTPUT
FILEDEF 07 DISK PASS DATA
FILEDEF 08 TERM
FILEDEF 09 DISK PASS HOLD
FILEDEF 10 DISK PASPLT DATA
CLRSCRN
&TYPE PASSIVE ACOUSTIC SEARCH SIMULATION *** "PASS"
& TYPE
&TYPE EXECUTING PROGRAM "PASS" ** "PASS" MUST BE FORTRAN-G
  OR FORTRAN-H
&TYPE PRECOMPILED ON YOUR A-DISK.  THE NONIMSL LIBRARY MUST
  BE AVAILABLE.
&TYPE
&TYPE DATA FILE MANAGEMENT:
&TYPE
&TYPE A. INPUT DATA FILE "PASS DATA":
&TYPE THIS FILE CONTAINS THE MOST RECENT DATA FOR RUNNING PASS.
&TYPE IT IS CREATED, OR MODIFIED, AFTER RUNNING THE INTERACTIVE
&TYPE DATA INPUT SECTION OF PASS.  SEE FURTHER NOTES ON OUTPUT
  DATA
&TYPE FILE "PASS HOLD".
&TYPE
&TYPE B. OUTPUT DATA FILE "PASS HOLD":
&TYPE UPON EXECUTIO OF PASS, IF FILE "PASS DATA" ALREADY
  EXISTS,
&TYPE IT IS COPIED INTO "PASS HOLD" UNCHANGED.  THUS IF YOU
  CHANGE THE
&TYPE INPUT DATA DURING THE INTERACTIVE SECTION, THE CHANGED
  DATA WILL
&TYPE BE IN "PASS DATA", AND THE UNCHANGED DATA WILL BE IN
  "PASS HOLD".
&TYPE
&TYPE C. OUTPUT DATA FILE "PASPLT DATA":
&TYPE THIS FILE CONTAINS ALL THE DATA, IN THE PROPER FORMAT,
  NEEDED
&TYPE TO GENERATE GRAPHICAL OUTPUT USING PROGRAM "PASSPLT",
  AND THE
```

```
&TYPE DISSPLA GRAPHICS SYSTEM.
&TYPE
&TYPE D. OUTPUT DATA FILE "PASS OUTPUT":
&TYPE THIS FILE CONTAINS THE STATISTICAL OUTPUT FROM THE
  MOST RECENT
&TYPE RUN OF PASS.
&TYPE
&TYPE ------------------------------------------------------------
LOAD PASS (START
```

APPENDIX H

FORTRAN CODE FOR PASS

```
C
C
C    *******************************************************************
C    *******************************************************************
C    ****                                                           ****
C    ****  PROGRAM "PASS" IS A MONTE-CARLO SIMULATION OF PASSIVE    ****
C    ****  SONAR ASW SEARCH EMPLOYING A THRESHOLD-CROSSING MODEL    ****
C    ****  TO DETERMINE IF DETECTION HAS OCCURRED, AND A "LAMBDA-   ****
C    ****  SIGMA-JUMP" OR A GAUSS-MARKOV STOCHASTIC ERROR PROCESS   ****
C    ****  TO MODEL FLUCTIONS IN ACOUSTIC SIGNAL EXCESS.            ****
C    ****                                                           ****
C    *******************************************************************
C    ****                                                           ****
C    ****  "PASS" WAS INSPIRED BY A SEARCH SIMULATION PROGRAM       ****
C    ****  WRITTEN BY L. K. ARNOLD OF DANIEL H. WAGNER ASSOC., (AS  ****
C    ****  DESCRIBED IN A MEMO FROM WAGNER ASSOC. TO COMSUBDEVRON  ****
C    ****  TWELVE OF 10 JUNE, 1975). SOME ADDITIONS AND IMPROVE-    ****
C    ****  MENTS INCORPORATED IN "PASS" WERE INSPIRED BY PROGRAM    ****
C    ****  "SEARCH", WHICH IS RESIDENT ON THE SUBDEVRON-12 SARF     ****
C    ****  COMPUTER. "SEARCH" WAS DEVELOPED BY W. J. BROWNING OF    ****
C    ****  APPLIED MATHEMATICS, INC. "PASS" ALSO INCLUDES MANY      ****
C    ****  IMPROVEMENTS, AND SOME CORRECTIONS TO THESE EARLIER      ****
C    ****  SEARCH SIMULATION PROGRAMS.                              ****
C    ****                                                           ****
C    *******************************************************************
C    ****                                                           ****
C    ****  EXTERNAL SUBROUTINES:                                    ****
C    ****                                                           ****
C    ****   NAME        LIBRARY     USE                             ****
C    ****                                                           ****
C    ****   FRTCMS      NONIMSL     ISSUE CMS COMMANDS FROM FORTRAN ****
C    ****   HISTGF      NONIMSL     HISTOGRAM AND STATISTICS        ****
C    ****   INTRPL      NONIMSL     INTERPOLATION                   ****
C    ****   LEXPN       NONIMSL     EXPONENTIAL RANDOM NUMBERS      ****
C    ****   LNORM       NONIMSL     NORMAL RANDOM NUMBERS           ****
C    ****   LRND        NONIMSL     UNIFORM RANDOM NUMBERS          ****
C    ****   SECTN       NONIMSL     SECTIONING DATA                 ****
C    ****   SHSORT      NONIMSL     SHELL SORT                      ****
C    ****                                                           ****
C    *******************************************************************
C    ****                                                           ****
C    ****  INTERNAL SUBROUTINES:                                    ****
C    ****                                                           ****
C    ****   NAME        USE                                         ****
C    ****                                                           ****
C    ****   CLEAR       CLEARS SCREEN                               ****
C    ****   DESCRB      BASIC PROGRAM DESCRIPTION TO SCREEN         ****
C    ****   ECHO1-6     DATA OUTPUT TO A-DISK (FILE 6)              ****
```

```
C     ****   EXPO           SINGLE EXPONENTIAL RANDOM NUMBER FROM STACK ****
C     ****   CPTNC-20       INTERACTIVE DATA INPUT AND SETUP            ****
C     ****   PAGE           OUTPUT PAGING                               ****
C     ****   READ IT        DATA INPUT FROM A-DISK (FILE 7)             ****
C     ****   SINKEM         OUTPUT RESULTS TO A-DISK (FILE 6)           ****
C     ****                  OUTPUT GRAPHICS DATA TO A-DISK (FILE 10)    ****
C     ****                  OUTPUT QUICK-SUMMARY TO SCREEN (FILE 8)     ****
C     ****   SKIP2          LINE SKIP FOR TERMINAL OUTPUT               ****
C     ****   UZ1            SINGLE UNIFORM RANDOM NUMBER FROM STACK     ****
C     ****   XDISTE         INITIAL TARGET X-POSITION FROM STACK        ****
C     ****   XLS            SINGLE NORMAL RANDOM NUMBER FROM STACK      ****
C     ****                                                              ****
C     ********************************************************************
C     ********************************************************************
C     ****                                                              ****
C     ****  ASSOCIATED PROGRAMS:                                        ****
C     ****                                                              ****
C     ****  PASS EXEC --- AN EXECUTIVE PROGRAM WHICH SETS UP INPUT-     ****
C     ****                OUTPUT FILES AND EXECUTES PROGRAM PASS.       ****
C     ****                                                              ****
C     ****  PASPLT FORTRAN --- A PROGRAM USED WITH THE DISSPLA          ****
C     ****                GRAPHICS SYSTEM TO PRODUCE GRAPHICAL          ****
C     ****                OUTPUT FROM DATA GENERATED BY PASS.           ****
C     ****                                                              ****
C     ********************************************************************
C     ****                                                              ****
C     ****  NPS VERSION REVISION HISTORY:                               ****
C     ****                                                              ****
C     ****    REVISION        PROGRAMMER              DATE              ****
C     ****                                                              ****
C     ****    ORIGINAL        LCDR S. G. SLATON       17 JUNE 1983      ****
C     ****                    (ADVISOR: J. EAGLE)                       ****
C     ****                                                              ****
C     ********************************************************************
C     ********************************************************************
C
C
      REAL XP(50),YP(50),RC(20),OL(20),RT(20),TL(20),DX(50),DY(50),
     &     DIST(50),NCZDO(5),NCZDT(5),NTDO(5000),NTDT(5000),TIFL(7),
     &     AFL(3),RCZO(5,2),RCZT(5,2),CZLO(5),CZLT(5),SPEED(2),TIME(2),
     &     OFOM(2),TFOM(2),RNTDO(5000),RNTDT(5000),STACK1(5000),STACK2(
     &     5000),STACK3(5000),UU(1),VV(1),SIGMA(3),ALAM(3),XODT(5000),YO
     &     DT(5000),XOTD(5000),YOTD(5000),XTDO(5000),YTDO(5000),XTDD(500
     &     0),YTDD(5000),FO(1000),TO(1000),FT(1000),TT(1000),PBIN(20)
C
      REAL XBIN(20),STACK4(5000),BRG(50),FOMBS(50),FOMBD(50)
C
      INTEGER NEXT(50),C1,C2,C3,C4,SEED
```

```
C
      LOGICAL AFLLS,AFLGM,FILE,ECHOS,STATS1,STATS2,GO4IT,ENUFF,ENUF,
     &        INTEG,SPECX,ASPECT,RFLCT
C
      EQUIVALENCE (TIFL(4),TCC),(TIFL(5),TTCC),(TIFL(6),TTSC),
     &            (SPEED(1),SOD),(SPEED(2),SOS),(TIME(1),TD),
     &            (TIME(2),TS),(OFOM(1),FOMOD),(OFOM(2),FOMOS),
     &            (TFOM(1),FOMTD),(TFOM(2),FOMTS),(TIFL(7),TSC)
C
      COMMON SEED,C1,C2,C3,C4
C
      DATA TWOPI/6.283185307/
C
C     +---------------------------------------------------------------+
C     |                                                               |
C     |             PART ONE (INITIALIZATION/DATA INPUT)              |
C     |                                                               |
C     |    1. INITIALIZE LOGICAL VARIABLES TO DEFAULT VALUES.         |
C     |    2. INPUT DATA BY:                                          |
C     |          A. INTERACTIVE SCREEN, OR                            |
C     |          B. EXISTING DATA FILE, OR                            |
C     |          C. COMBINATION OF A AND B                            |
C     |       DEPENDING ON OPTIONS SELECTED.                          |
C     |                                                               |
C     +---------------------------------------------------------------+
C
      FILE=.TRUE.
      AFLGM=.FALSE.
      AFLLS=.TRUE.
      ECHOS=.TRUE.
      STATS1=.TRUE.
      STATS2=.TRUE.
      GO4IT=.FALSE.
      ENUFF=.FALSE.
      ENUF=.FALSE.
      INTEG=.FALSE.
      SPECX=.FALSE.
      ASPECT=.FALSE.
      RFLCT=.FALSE.
C
      CALL DESCRE
      CALL CLEAR
      CALL OPTN0(GO4IT)
      IF (GO4IT) GO TO 5
C
      CALL OPTN2A(INTEG)
      CALL OPTN2(AFLLS,AFLGM)
      CALL OPTN3A(FILE)
```

```
      CALL OPTN3(XMAX,YMAX,FILE)
      CALL OPTN4(NP,KP,XP,YP,FILE)
      CALL OPTN5(MO,RO,CL,FILE)
      CALL OPTN6(MT,RT,TL,FILE)
      CALL OPTN7(NCZO,RCZO,CZLO,FILE)
      CALL OPTN8(NCZT,RCZT,CZLT,FILE)
      CALL OPTN9(KBAR,START,ANG2,FILE,SPECX)
      IF (SPECX) CALL OPTN9B(XMAX,NBINS,PBIN,XBIN)
      CALL OPTN10(FOMOD,FOMOS,FILE,ASPECT)
      IF (ASPECT) CALL OPT10B(NLS,BRG,FOMBS,FOMBD)
      CALL OPTN11(SOD,SOS,FILE)
      CALL OPTN12(TD,TS,FILE)
      CALL OPTN13(FOMTD,FOMTS,FILE)
      CALL OPTN14(STMIN,STMAX,FILE)
      CALL OPTN15(RTSC,RTCC,FILE)
      CALL OPTN16(ALAM,FILE)
      CALL OPTN17(SIGMA,FILE)
      CALL OPTN18(SEED,NREP,TMAX,FILE)
      CALL OPTN19(ECHOS)
      CALL OPTN20(STATS1,STATS2)
      CALL CLEAR
      GO TO 6
    5 CALL CLEAR
      CALL READIT(XMAX,YMAX,NP,KP,XP,YP,MO,RO,CL,MT,RT,TL,NCZO,RCZO,CZLO
     *,NCZT,RCZT,CZLT,KBAR,START,ANG2,FOMOD,FOMOS,SOD,SOS,TD,TS,FOMTD,
     *FOMTS,STMIN,STMAX,RTSC,RTCC,ALAM,SIGMA,SEED,NREP,TMAX)
    6 CONTINUE
C
C     +-----------------------------------------------------------------+
C     |                                                                 |
C     |               PART TWO (DATA OUTPUT/INITIALIZATION)             |
C     |                                                                 |
C     |   1. SENDS INPUT DATA TO A-DISK OUTPUT FILE, IF USER SO         |
C     |      SPECIFIES IN OPTIONS.                                      |
C     |   2. INITIALIZES MAXIMUM TARGET COURSE VARIATION IF BARRIER     |
C     |      SCENARIO SELECTED BY USER IN OPTIONS.                      |
C     |                                                                 |
C     +-----------------------------------------------------------------+
C
      IF (.NOT. ECHOS) GO TO 9210
C
      CALL ECHO1(XMAX,YMAX,NP,KP,XP,YP,MO,RO,CL,MT,RT,TL)
C
      IF (NCZO .GT. 0) CALL ECHO2(NCZO,RCZO,CZLO)
      IF (NCZT .GT. 0) CALL ECHO3(NCZT,RCZT,CZLT)
C
      CALL ECHO4(KBAR)
 9210 IF (KBAR .LE. 1) GO TO 921
```

```
      IF (ECHOS) CALL ECHO5(ANG2,START)
      ANG2=ANG2*(TWOPI/360.0)
      ANG1=(3.0)*(TWOPI/4.0)-ANG2
      ANG2=(2.0)*ANG2
  921 CONTINUE
      IF (.NOT. ECHOS) GO TO 9211
      CALL ECHO6(FOMOD,FOMOS,SOD,SOS,TD,TS,FOMTD,FOMTS,STMIN,STMAX,RTSC,
     *            RTCC,SEED,NREP,IMAX,SIGMA,ALAM,AFLGM,AFLLS,INTEG,ASPECT
     *            ,NLS,BRG,FOMBS,FOMBD)
 9211 CONTINUE
C
C     +-----------------------------------------------------------------+
C     |                                                                 |
C     |            PART THREE (INITIALIZATION AND SET-UP)               |
C     |                                                                 |
C     |   1. INITIALIZES DETECTION COUNTERS.                            |
C     |   2. INITIALIZES POINTERS FOR RETRIEVAL OF RANDOM NUMBERS       |
C     |      FROM APPROPRIATE STACKS.                                   |
C     |   3. SETS-UP SEARCHER PATH DIRECTIONS AND DISTANCES FOR EACH    |
C     |      LEG OF THE SPECIFIED SEARCH PATTERN.                       |
C     |   4. SETS-UP TARGET SPEED INCREMENT AND VARIABLES USED IN       |
C     |      BOUNDARY REFLECTION ROUTINE.                               |
C     |                                                                 |
C     +-----------------------------------------------------------------+
C
      J=NP-1
      DO 200 I=1,J
         NEXT(I)=I+1
  200 CONTINUE
      NEXT(NP)=KP
      DO 201 I=1,NP
         J=NEXT(I)
         X=XP(J)-XP(I)
         Y=YP(J)-YP(I)
         D=SQRT((X*X)+(Y*Y))
         DX(I)=X/D
         DY(I)=Y/D
         DIST(I)=D
  201 CONTINUE
      NDC=0
      NDT=0
      DO 203 I=1,5
         NCZCO(I)=0
         NCZCT(I)=0
  203 CONTINUE
      NBOTH=0
      NONE=0
      TOXMAX=2.0*XMAX
```

```
      TCYMAX=2.0*YMAX
      STINC=STMAX-STMIN
      C1=5000
      C2=5000
      C3=5000
      C4=0
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |             PART FOUR (REPLICATION DO-LOOP START)               |
C    |                                                                 |
C    |   1. START OF THE "DO FOR" LOOP ON THE NUMBER OF REPLICATIONS.  |
C    |   2. INITIALIZES VARIABLES FOR EACH REPLICATION.                |
C    |   3. OUTPUTS REPLICATION COUNT TO SCREEN EVERY 200 REPS.        |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
      CALL CLEAR
         KTEST=200
         CALL SKIP2
         WRITE(8,100)
      DO 300 KREP=1,NREP
         IF (KREP .EQ. KTEST) GO TO 999
         GO TO 998
  999    WRITE(8,*)KREP,NREP
         KTEST=KTEST+200
  998    CONTINUE
         IF (.NOT. ENUFF) M1=0
         IF (.NOT. ENUF ) M2=0
         TNOW=0.0
         TLAST=0.0
         MSELO=0
         MSELT=0
         KCZC=0
         KCZT=0
         J=0
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |                     PART FIVE (SET-UP)                          |
C    |                                                                 |
C    |   1. SET-UP FOR START OF SIMULATION:                            |
C    |       A. ACOUSTIC FLUCTUATION LEVELS/TIME OF NEXT LEVEL CHANGE. |
C    |       B. SEARCHER AND TARGET SONAR FIGURE-OF-MERIT.             |
C    |       C. SEARCHER START POSITION.                               |
C    |       D. SEARCH MODE (ALWAYS STARTS IN "DRIFT").                |
C    |       E. SEARCH SPEED/TIME TO SEARCHER SPEED CHANGE.            |
C    |       F. SEARCHER LEG DISTANCE.                                 |
```

```
C     |        G. TIME TO SEARCHER COURSE CHANGE.                               |
C     |        H. X,Y COMPONENTS OF SEARCHER VELOCITY.                          |
C     |        I. INITIAL TARGET POSITION AND SPEED.                            |
C     |        J. TIME TO TARGET SPEED CHANGE.                                  |
C     |        K. INITIAL TARGET COURSE.                                        |
C     |        L. X,Y COMPONENTS OF TARGET VELOCITY.                            |
C     |        M. TIME TO TARGET COURSE CHANGE.                                 |
C     |                                                                         |
C     +-------------------------------------------------------------------------+
C
      DO 301 I=1,3
         CALL EXPO(ALAM(I),T,STACK1)
         TIFL(I)=T
         CALL XLS(SIGMA(I),X,STACK2)
         AFL(I)=X
  301 CONTINUE
      XO=XP(1)
      YO=YP(1)
      NLEG=1
      MODE=1
      TSC=TIME(MODE)
      FOMT=TFOM(MODE)
      FOMO=OFOM(MODE)
      SO=SPEED(MODE)
      D=DIST(1)
      TOO=D/SO
      VX=DX(1)*SO
      VY=DY(1)*SO
      IF (SPECX) GO TO 8081
      CALL UZ1(U,STACK3)
      XT=U*XMAX
      GO TO 8082
 8081 CALL XDISTB(NBINS,PBIN,XBIN,NREP,XT,STACK4)
 8082 CONTINUE
      CALL UZ1(U,STACK3)
      GO TO (807,808),KBAR
  808 YT=YMAX-U*(START*(STMIN+STMAX)/2.0)
      GO TO 809
  807 YT=U*YMAX
  809 CONTINUE
      CALL UZ1(U,STACK3)
      ST=STMIN+U*STINC
      CALL EXPO(FTSC,T,STACK1)
      TTSC=T
      CALL UZ1(U,STACK3)
      GO TO (810,811),KBAR
  811 X=ANG1+U*ANG2
      GO TO 812
```

```
 810 X=L*TWOPI
 812 IF (ASPECT) THETA=X
     COSX=COS(X)
     SINX=SIN(X)
     UX=ST*COSX
     UY=ST*SINX
     CALL EXPO(RTOO,T,STACK1)
     TTOO=T
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |             PART SIX (DETERMINE DETECTION RANGES)               |
C    |                                                                 |
C    |   1. CALCULATES DIRECT PATH (DP) DETECTION RANGES AND          |
C    |      CONVERGENCE ZONE (CZ) DETECTION POSSIBILITIES BASED       |
C    |      ON CURRENT FIGUER-OF-MERIT AND FLUCTUATION LEVELS.        |
C    |   2. STORES FIVE HOURS OF FIGURE-OF-MERIT DATA FOR GRAPHICAL   |
C    |      PRESENTATION.                                              |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
 310 CONTINUE
     IF (.NOT. ASPECT) GO TO 3101
        CALL RELE(TWOPI,THETA,XO,YO,XT,YT,BREL)
        UU(1)=BREL
        GO TO (3102,3103),MODE
3102 CALL INTRPL(NLS,BRG,FOMBD,1,UU,VV)
     GO TO 3104
3103 CALL INTRPL(NLS,BRG,FOMBS,1,UU,VV)
3104 FOMO=VV(1)
3101 X=FOMO+AFL(1)+AFL(2)
     IF (ENUFF) GO TO 312
     M1=M1+1
     FO(M1)=X
     TO(M1)=TNOW
     IF (TO(M1) .GE. 5.0) ENUFF=.TRUE.
 312 CONTINUE
     UU(1)=X
 311 CALL INTRPL(MO,OL,RO,1,UU,VV)
     RNGO=VV(1)
     IF (NCZO .EQ. 0) GO TO 360
     DO 361 I=1,NCZO
        IF (X .GE. CZLO(I)) GO TO 361
        KCZO=I-1
        GO TO 360
 361 CONTINUE
     KCZO=NCZO
 360 CONTINUE
```

```
  315 X=FOMT+AFL(2)+AFL(3)
      IF (ENUF) GO TO 314
      M2=M2+1
      FT(M2)=X
      TT(M2)=TNOW
      IF (TT(M2) .GE. 5.0) ENUF=.TRUE.
  314 CONTINUE
      UU(1)=X
  313 CALL INTRPL(MT,TL,RT,1,UU,VV)
      RNGT=VV(1)
      IF (NCZT .EQ. 0) GO TO 362
      DO 363 I=1,NCZT
         IF (X .GE. CZLT(I)) GO TO 363
         KCZT=I-1
         GO TO 362
  363 CONTINUE
      KCZT=NCZT
  362 CONTINUE
      RMAX=AMAX1(RNGO,RNGT)
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |           PART SEVEN (CHECK FOR DETECTION CRITERIA)             |
C    |                                                                 |
C    |  1. COMPARE DP DETECTION RANGES TO ACTUAL RANGE TO DETERMINE    |
C    |     IF DP DETECTION CRITERIA MET.                               |
C    |  2. CHECKS RANGES TO DETERMINE IF CZ DETECTION CRITERIA MET.   |
C    |  3. TRANSFER TO PART 16-A IF ANY DETECTION/COUNTERDETECTION    |
C    |     CRITERIA MET.                                               |
C    |  4. CONTINUE TO PART 8 IF NO DETECTION CRITERIA MET.           |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
  320 RNG=SQRT(((XO-XT)**2)+((YO-YT)**2))
      TBIG=RNG-RMAX
      IF (KCZO .EQ. 0) GO TO 364
      DO 365 I=1,KCZO
         ICZO=KCZO-I+1
         IF (RNG .GT. RCZO(ICZO,2)) GO TO 366
         IF (RNG .GE. RCZO(ICZO,1)) GO TO 367
  365 CONTINUE
      GO TO 364
  366 TBIG=AMIN1(TBIG,RNG-RCZO(ICZO,2))
      GO TO 364
  367 KCZO=-1
  364 IF (KCZT .EQ. 0) GO TO 368
      DO 369 I=1,KCZT
         ICZT=KCZT-I+1
```

```
      IF (RNG .GT. RCZT(ICZT,2)) GO TO 370
      IF (RNG .GE. RCZT(ICZT,1)) GO TO 371
  369 CONTINUE
      GO TO 368
  370 TBIG=AMIN1(TBIG,RNG-RCZT(ICZT,2))
      GO TO 368
  371 KCZT=-1
  368 IF (KCZO .LT. 0 .OR. KCZT .LT. 0 .OR. TBIG .LE. 0) GO TO 330
C
C     +----------------------------------------------------------------------+
C     |                                                                      |
C     |  PART EIGHT (DETERMINE TIME INCREMENT BASED ON "NEXT EVENT")         |
C     |                                                                      |
C     |   1. IF SIGNAL INTEGRATION IN EFFECT, DECREMENT MSEL COUNTERS        |
C     |      THAT ARE GREATER THAN ZERO.                                     |
C     |   2. SET TIME INCREMENT TO 3 MINUTES IF ANY MSEL COUNTER IS          |
C     |      GREATER THAN ZERO.                                              |
C     |   3. CHECK ALL EVENTS FROM FOLLOWING LIST, SET TIME INCREMENT        |
C     |      TO EARLIEST EVENT.                                              |
C     |       A. ACCUSTIC FLUCTUATION LEVEL CHANGE (J=1,2,3).                |
C     |       B. SEARCHER COURSE CHANGE (J=4).                               |
C     |       C. TARGET COURSE CHANGE (J=5).                                 |
C     |       D. TARGET SPEED CHANGE (J=6).                                  |
C     |       E. SEARCHER SPEED CHANGE (J=7).                                |
C     |       F. NONE OF THE ABOVE (J=8).                                    |
C     |   4. END REPLICATION IF MAXIMUM TIME REACHED.                        |
C     |                                                                      |
C     +----------------------------------------------------------------------+
C
      IF (.NOT. INTEG) GO TO 3681
      IF (MSELO .GT. 0) MSELO=MSELO-1
      IF (MSELT .GT. 0) MSELT=MSELT-1
 3680 IF (MSELO .GT. 0 .OR. MSELT .GT. 0) TBIG=0.0
 3681 TBIG=AMAX1(TBIG/(SO+ST),0.05)
      TINC=TBIG+TNOW
      J=8
      DO 321 I=1,7
         IF (TINC .LT. TIFL(I)) GO TO 321
         TINC=TIFL(I)
         J=I
  321 CONTINUE
      IF (TINC .GE. TMAX) GO TO 340
C
C     +----------------------------------------------------------------------+
C     |                                                                      |
C     |                     PART NINE-A (MOVE THE SHIPS)                     |
C     |                                                                      |
C     |   1. MOVE THE SEARCHER ALONG TRACK A DISTANCE CORRESPONDING TO       |
```

```
C       |     SEARCHER SPEED AND TIME INCREMENT, CHECKING FOR DISTANCE        |
C       |     AND TIME TO SEARCHER COURSE CHANGE BASED ON NEXT LEG OF         |
C       |     SEARCHER TRACK.                                                 |
C       |  2. MOVE THE TARGET IN A SIMILAR MANNER.                            |
C       |  3. INCREMENT PROBLEM TIME.                                         |
C       |                                                                     |
C       +---------------------------------------------------------------------+
C
        X=TINC-TNOW
        XOT=XO+(VX*X)
        YOT=YO+(VY*X)
        NL=NEXT(NLEG)
        D1=SQRT(((XOT-XO)**2)+((YOT-YO)**2))
        D2=SQRT(((XO-XP(NL))**2)+((YO-YP(NL))**2))
        IF (D1 .LE. D2) GO TO 3211
              X=(D2/SO)
              J=4
              GO TO 3212
 3211   CONTINUE
        XO=XOT
        YO=YOT
 3212   XT=XT+(UX*X)
        YT=YT+(UY*X)
        TNOW=TNOW+X
C
C       +---------------------------------------------------------------------+
C       |                                                                     |
C       |              PART NINE-B (TARGET BOUNDARY REFLECTION)               |
C       |                                                                     |
C       |  1. CHECK TARGET INSIDE SEARCH BOUNDARIES. IF OUTSIDE,              |
C       |     "REFLECT" POSITION IN AND "REFLECT" APPROPRIATE TARGET          |
C       |     VELOCITY VECTORS.                                               |
C       |  2. IN BARRIER SCENARIO, IF TARGET HAS CROSSED X-AXIS,              |
C       |     END THE REPLICATION.                                            |
C       |                                                                     |
C       +---------------------------------------------------------------------+
C
        RFLCT=.FALSE.
  326   IF (XT) 322,323,323
  322   RFLCT=.TRUE.
        XT=-XT
        UX=-UX
        COSX=-COSX
  323   IF (XT-XMAX) 324,324,325
  325   RFLCT=.TRUE.
        XT=TOXMAX-XT
        UX=-UX
        COSX=-COSX
```

```
      GO TO 326
  324 IF (YT) 327,328,328
  327 RFLCT=.TRUE.
      YT=-YT
      IF (KBAR .EQ. 2) GO TO 340
      UY=-UY
      SINX=-SINX
  328 IF (YT-YMAX) 3291,3291,3290
 3290 RFLCT=.TRUE.
      YT=TOYMAX-YT
      UY=-UY
      SINX=-SINX
      GO TO 324
 3291 IF (.NOT. (ASPECT .AND. RFLCT)) GO TO 329
         IF (SINX .GE. 0.0 .AND. COSX .GE. 0.0) GO TO 3293
         IF (COSX .LT. 0.0) GO TO 3294
         THETA=TWOPI+ARSIN(SINX)
         GO TO 329
 3293    THETA=ARSIN(SINX)
         GO TO 329
 3294    THETA=(TWOPI/2.0)-ARSIN(SINX)
         GO TO 329
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |              PART TEN (BRANCH TO "NEXT EVENT")                 |
C     |                                                                |
C     |   1. BRANCH ON THE VALUE OF J, WHICH WAS DETERMINED IN PART    |
C     |      EIGHT, AND WHICH "POINTS" TO THE PROGRAM PART WHICH      |
C     |      CORRESPONDS TO THE "NEXT EVENT".                          |
C     |                                                                |
C     +----------------------------------------------------------------+
C
  329 IF (.NOT. INTEG) GO TO 3292
      IF (J .EQ. 8 .AND. KOZO .LT. 0) GO TO 310
      IF (J .EQ. 8 .AND. RFLCT .AND. ASPECT) GO TO 310
      IF (J .EQ. 8 .AND. KOZT .LT. 0) GO TO 315
 3292 GO TO (350,350,350,351,352,354,353,320),J
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |         PART ELEVEN (FLUCTUATION LEVEL CHANGE/J=1,2,3/)        |
C     |                                                                |
C     |   1. DEPENDING ON THE ACOUSTIC FLUCTUATION MODEL CHOSEN,       |
C     |      CALCULATE NEW VALUE OF THE STOCHASTIC ERROR PROCESS,     |
C     |      THEN RETURN TO PART SIX.                                  |
C     |                                                                |
C     +----------------------------------------------------------------+
```

```
C
C
  350 IF (AFLGM) GO TO 3501
      CALL XLS(SIGMA(J),X,STACK2)
      AFL(J)=X
      CALL EXPO(ALAM(J),T,STACK1)
      TIFL(J)=T+TNOW
      IF (RFLCT .AND. ASPECT) GO TO 310
      GO TO (310,310,315),J
 3501 CONTINUE
      CALL EXPO(ALAM(J),T,STACK1)
      TIFL(J)=TNOW+T
 3503 CONTINUE
      S=TNOW-TLAST
      TLAST=TNOW
      DO 3502 JJ=1,3
         CALL XLS(SIGMA(JJ),X,STACK2)
         ALFA=EXP(-(ALAM(JJ)*S))
         BETA=SQRT(1.0-ALFA**2)
         AFL(JJ)=ALFA*AFL(JJ)+BETA*X
 3502 CONTINUE
      GO TO 310
C
C     +----------------------------------------------------------------------+
C     |                                                                      |
C     |              PART TWELVE (SEARCHER COURSE CHANGE/J=4/)               |
C     |                                                                      |
C     |    1. FIX SEARCHER POSITION AT THE START OF THE NEXT LEG.            |
C     |    2. COMPUTE NEW VELOCITY VECTORS, LEG DISTANCE, AND TIME OF        |
C     |       NEXT SEARCHER COURSE CHANGE.                                   |
C     |    3. RETURN TO PART SIX OR SEVEN, DEPENDING ON CURRENT              |
C     |       DETECTION CRITERIA.                                            |
C     |                                                                      |
C     +----------------------------------------------------------------------+
C
  351 NLEG=NEXT(NLEG)
      XC=XP(NLEG)
      YC=YP(NLEG)
      VX=DX(NLEG)*SO
      VY=DY(NLEG)*SO
      D=DIST(NLEG)
      TCC=TNOW+D/SO
      IF (RFLCT .AND. ASPECT) GO TO 310
      IF (.NOT. INTEG) GO TO 320
      IF (KCZO .LT. 0) GO TO 310
      IF (KCZT) 315,320,320
C
C     +----------------------------------------------------------------------+
```

```
C     +------------------------------------------------------------------+
C     |                                                                  |
C     |            PART THIRTEEN (TARGET COURSE CHANGE/J=5/)             |
C     |                                                                  |
C     |   1. CALCULATE NEW VELOCITY VECTORS AND TIME TO TARGET COURSE    |
C     |      CHANGE.                                                     |
C     |   2. RETURN TO PART SIX OR SEVEN DEPENDING ON PRESENT DETECTION  |
C     |      CRITERIA, OR PART ELEVEN IF GAUSS-MARKOV AND ASPECT USED.   |
C     |                                                                  |
C     +------------------------------------------------------------------+
C
  352 CALL UZ1(U,STACK3)
      GO TO (813,814),KBAR
  814 X=ANG1+U*ANG2
      GO TO 815
  813 X=U*TWOPI
  815 IF (ASPECT) THETA=X
      COSX=COS(X)
      SINX=SIN(X)
      UX=ST*COSX
      UY=ST*SINX
      CALL EXPO(RTCC,T,STACK1)
      TTCC=TNOW+T
      IF (ASPECT .AND. AFLGM) GO TO 3503
      IF (ASPECT) GO TO 310
      IF (.NOT. INTEG) GO TO 320
      IF (KCZO .LT. 0) GO TO 310
      IF (KCZT) 315,320,320
C
C     +------------------------------------------------------------------+
C     |                                                                  |
C     |            PART FOURTEEN (SEARCHER SPEED CHANGE/J=7/)            |
C     |                                                                  |
C     |   1. CHANGE SEARCHER SPEED (SPRINT/DRIFT).                       |
C     |   2. CALCULATE TIME TO NEXT SEARCHER SPEED CHANGE.               |
C     |   3. CALCULATE TIME TO NEXT SEARCHER COURSE CHANGE BASED ON      |
C     |      NEW SPEED AND DISTANCE LEFT ON CURRENT LEG.                 |
C     |   4. RETURN TO PART ELEVEN IF FLUCTUATION MODEL IS GAUSS-        |
C     |      MARKOV, OTHERWISE RETURN TO PART SIX.                       |
C     |                                                                  |
C     +------------------------------------------------------------------+
C
  353 MODE=3-MODE
      FOMO=OFOM(MODE)
      FOMT=TFOM(MODE)
      SO=SPEED(MODE)
      TSC=TNOW+TIME(MODE)
      NL=NEXT(NLEG)
      D=SQRT(((XO-XP(NL))**2)+((YO-YP(NL))**2))
```

```
      TCC=TNOW+D/SC
      IF (AFLGM) GO TO 3503
      GO TO 310
C
C     +-----------------------------------------------------------------+
C     |                                                                 |
C     |            PART FIFTEEN (TARGET SPEED CHANGE/J=6/)              |
C     |                                                                 |
C     |   1. CALCULATE NEW TARGET SPEED VECTORS AND TIME TO NEXT TARGET |
C     |      SPEED CHANGE.                                              |
C     |   2. RETURN TO PART SIX OR SEVEN DEPENDING ON PRESENT DETECTION |
C     |      CRITERIA.                                                  |
C     |                                                                 |
C     +-----------------------------------------------------------------+
C
  354 CALL UZ1(U,STACK3)
      ST=STMIN+U*STINC
      CALL EXPO(RTSC,T,STACK1)
      TTSC=TNOW+T
      UX=ST*COSX
      UY=ST*SINX
      IF (RFLCT .AND. ASPECT) GO TO 310
      IF (.NOT. INTEG) GO TO 320
      IF (KCZO .LT. 0) GO TO 310
      IF (KCZT) 315,320,320
C
C     +-----------------------------------------------------------------+
C     |                                                                 |
C     |            PART SIXTEEN-A (DETERMINE WHO DETECTS)               |
C     |                                                                 |
C     |   1. TRANSFER TO SEARCHER DETECTION, TARGET DETECTION, OR       |
C     |      SIMULTANEOUS DETECTION, AS APPROPRIATE.                    |
C     |                                                                 |
C     +-----------------------------------------------------------------+
C
  330 IF (RNGT .LT. RNG .AND. KCZT .GE. 0) GO TO 333
      IF (RNGO .LT. RNG .AND. KCZO .GE. 0) GO TO 334
C
C     +-----------------------------------------------------------------+
C     |                                                                 |
C     |            PART SIXTEEN-B (SIMULTANEOUS DETECTION)              |
C     |                                                                 |
C     |   1. INCREMENT BOTH MSEL COUNTERS, IF APPROPRIATE.              |
C     |   2. CALL SIMULTANEOUS DETECTION, IF APPROPRIATE, AND END       |
C     |      REPLICATION.                                               |
C     |   3. IF DETECTION CRITERIA NOT MET, RETURN TO PART EIGHT.       |
C     |                                                                 |
C     +-----------------------------------------------------------------+
```

```
C
      IF (.NOT. INTEG) GO TO 3301
      MSELO=MSELO+1
      MSELT=MSELT+1
      IF (MSELO .GE. 3 .AND. MSELT .GE. 3) GO TO 3301
      IF (MSELO .GE. 3) GO TO 3331
      IF (MSELT .GE. 3) GO TO 3341
      GO TO 3680
 3301 NBOTH=NBOTH+1
      GO TO 300
C
C     +---------------------------------------------------------------+
C     |                                                               |
C     |        PART SIXTEEN-C (SEARCHER SECURE DETECTION)             |
C     |                                                               |
C     |   1. INCREMENT SEARCHER MSEL COUNTER AND DECREMENT TARGET     |
C     |      MSEL COUNTER, IF APPROPRIATE.                            |
C     |   2. CALL SEARCHER DETECTION, IF APPROPRIATE, AND END         |
C     |      REPLICATION.                                             |
C     |        A. INCREMENT NDC COUNTER.                              |
C     |        B. SAVE TIME, RANGE, AND POSITION DATA.                |
C     |   3. RETURN TO PART EIGHT IF MSEL CRITERIA NOT MET, IF        |
C     |      APPROPRIATE.                                             |
C     |                                                               |
C     +---------------------------------------------------------------+
C
  333 IF (.NOT. INTEG) GO TO 3331
      MSELO=MSELO+1
      IF (MSELO .GE. 3) GO TO 3331
      IF (MSELT .GT. 0) MSELT=MSELT-1
      GO TO 3680
 3331 NDC=NDC+1
      IF (KCZO .LT. 0) NCZDO(ICZO)=NCZDO(ICZO)+1.0
      NTDO(NDO)=TNOW
      RNTDO(NDO)=RNG
      XODT(NDO)=XO
      YODT(NDO)=YO
      XTDO(NDO)=XT
      YTDO(NDO)=YT
      GO TO 300
C
C     +---------------------------------------------------------------+
C     |                                                               |
C     |        PART SIXTEEN-D (TARGET SECURE DETECTION)               |
C     |                                                               |
C     |   1. INCREMENT TARGET MSEL COUNTER AND DECREMENT SEARCHER     |
C     |      MSEL COUNTER, IF APPROPRIATE.                            |
C     |   2. CALL TARGET DETECTION, IF APPROPRIATE, AND END           |
```

```
C       |       REPLICATION.                                                     |
C       |         A. INCREMENT NDT COUNTER.                                      |
C       |         B. SAVE TIME, RANGE, AND POSITION DATA.                        |
C       |    3. RETURN TO PART EIGHT IF MSEL CRITERIA NOT MET, IF                |
C       |       APPROPRIATE.                                                     |
C       |                                                                        |
C       +------------------------------------------------------------------------+
C
  334 IF (.NOT. INTEG) GO TO 3341
      MSELT=MSELT+1
      IF (MSELT .GE. 3) GO TO 3341
      IF (MSELO .GT. 0) MSELO=MSELO-1
      GO TO 3680
 3341 NDT=NDT+1
      IF (KCZT .LT. 0) NCZDT(ICZT)=NCZDT(ICZT)+1.0
      ATDT(NDT)=TNOW
      RNTDT(NDT)=RNG
      XTDO(NDT)=XT
      YTDO(NDT)=YT
      XCTD(NDT)=XC
      YCTD(NDT)=YC
      GO TO 300
C
C       +------------------------------------------------------------------------+
C       |                                                                        |
C       |                  PART SEVENTEEN (NO DETECTION)                         |
C       |                                                                        |
C       |    1. INCREMENT NONE COUNTER AND END REPLICATION.                      |
C       |                                                                        |
C       +------------------------------------------------------------------------+
C
  340 NONE=NONE+1
  300 CONTINUE
C
C       +------------------------------------------------------------------------+
C       |                                                                        |
C       |                  PART EIGHTEEN (RESULTS OUTPUT)                        |
C       |                                                                        |
C       |    1. RESULTS OF SIMULATION (TIME, RANGE) SENT TO A-DISK               |
C       |       (FILE 6) IN THE FORM OF HISTOGRAMS AND STATISTICS                |
C       |       (PRINTING FILE-6 PROVIDES HARDCOPY OUTPUT).                      |
C       |    2. RESULTS OF SIMULATION (CUMULATIVE PD, POSITIONS, ETC)            |
C       |       SENT TO A-DISK (FILE 10) AS DATA FOR "DISSPLA" GRAPHICS          |
C       |       SYSTEM. THIS DATA IS FORMATTED FOR USE WITH FORTRAN              |
C       |       PROGRAM "PASPLT" AND TEKTRONIX 618 GRAPHICS TERMINAL.            |
C       |    3. SIMULATION QUICKSUMMARY SENT TO TERMINAL SCREEN (FILE 8)         |
C       |       AND TO A-DISK (FILE 6).                                          |
C       |                                                                        |
```

```
C     |   NOTE: ALL ABOVE DONE BY CALLING SUBROUTINE "SINKEM".              |
C     |                                                                     |
C     +---------------------------------------------------------------------+
C
      CALL CLEAR
C
      CALL SINKEM(NDO,NTDO,RNTDO,NDT,NTDT,RNTDT,NONE,NREP,NCZDO,NCZDT,NC
     *ZD,NCZT,NBOTH,XODT,YODT,XTOD,YTOD,XTDO,YTDO,XOTD,YOTD,XMAX,YMAX,
     *STATS1,STATS2,KBAR,FC,TO,FT,TT,M1,M2,AFLLS)
      STOP
  100 FORMAT(2X,'THE SIMULATION IS UNDERWAY. PROGRESS MAY BE MONITORED B
     *Y WATCHING:',//,'  COMPLETED REPLICATIONS OF TOTAL REPLICATIONS')
      END
C
C     +---------------------------------------------------------------------+
C     |                                                                     |
C     |                        END OF MAIN PROGRAM                          |
C     |                                                                     |
C     +---------------------------------------------------------------------+
C
C
C
C     ***********************************************************************
C     ***********************************************************************
C
C
C
C     +---------------------------------------------------------------------+
C     |                                                                     |
C     |                         SUBROUTINE ECHO1                            |
C     |                                                                     |
C     +---------------------------------------------------------------------+
C
      SUBROUTINE ECHO1(XMAX,YMAX,NP,KP,XP,YP,MO,RO,OL,MT,RT,TL)
C
      REAL XP(NP),YP(NP),RO(MO),OL(MO),RT(MT),TL(MT),U(150),V(150),
     &XL(4),YL(4)
C
      CALL PAGE
      WRITE(6,99)
      WRITE(6,104)
      WRITE(6,100)XMAX, YMAX
C
      WRITE(6,105)NP,KP
      DO 10  I=1,NP
         WRITE(6,101)I,XP(I),I,YP(I)
   10 CONTINUE
C
      CALL PAGE
```

```
      WRITE(6,106)
      DO 11  I=1,NC
         WRITE(6,102)I,RC(I),I,OL(I)
   11 CONTINUE
C
      WRITE(6,107)
      DO 12  I=1,NT
         WRITE(6,103)I,RT(I),I,TL(I)
   12 CONTINUE
C
      RETURN
   99 FORMAT(//,' ECHO OF INPUT DATA FOLLOWS:')
  100 FORMAT(5X,'XMAX=',F7.2,/,5X,'YMAX=',F7.2)
  101 FORMAT(5X,'XP(',I2,')=',F7.2,5X,'YP(',I2,')=',F7.2)
  102 FORMAT(5X,'RO(',I2,')=',F7.2,5X,'OL(',I2,')=',F7.2)
  103 FORMAT(5X,'RT(',I2,')=',F7.2,5X,'TL(',I2,')=',F7.2)
  104 FORMAT(/,' SEARCH AREA DIMENSIONS:')
  105 FORMAT(//,' SEARCHER TRACK ANCHOR POINTS:',/,5X,'NP=',I2,5X,'KP=',
     *I2)
  106 FORMAT(//,' SEARCHER PROPAGATION LOSS:')
  107 FORMAT(//,' TARGET PROPAGATION LOSS:')
      END
C
C
C     ************************************************************************
C     ************************************************************************
C
C
C     +----------------------------------------------------------------------+
C     |                                                                      |
C     |                          SUBROUTINE ECHO2                            |
C     |                                                                      |
C     +----------------------------------------------------------------------+
C
      SUBROUTINE ECHO2(N,X,Y)
C
      REAL  X(5,2),Y(5)
C
      WRITE(6,101)
      DO 10  I=1,N
         WRITE(6,100)I,X(I,1),I,X(I,2),I,Y(I)
   10 CONTINUE
C
      RETURN
  100 FORMAT(5X,'RCZO(',I1,'1)=',F7.2,5X,'RCZO(',I1,'2)=',F7.2,5X,'CZLC(
     *',I1,')=',F7.2)
  101 FORMAT(//,' SEARCHER CONVERGENCE ZONES:')
      END
```

```
C
C
C     ********************************************************
C     ********************************************************
C
C
C     +------------------------------------------------------+
C     |                                                      |
C     |                  SUBROUTINE ECHO3                    |
C     |                                                      |
C     +------------------------------------------------------+
C
C
      SUBROUTINE ECHO3(N,X,Y)
C
      REAL X(5,2),Y(5)
C
      WRITE(6,101)
      DO 10 I=1,N
         WRITE(6,100)I,X(I,1),I,X(I,2),I,Y(I)
   10 CONTINUE
C
      RETURN
  100 FORMAT(5X,'RCZT(',I1,'1)=',F7.2,5X,'RCZT(',I1,'2)=',F7.2,5X,'CZLT(
     *',I1,')=',F7.2)
  101 FORMAT(//,' TARGET CONVERGENCE ZONES:')
      END
C
C
C     ********************************************************
C     ********************************************************
C
C
C
C     +------------------------------------------------------+
C     |                                                      |
C     |                  SUBROUTINE ECHO4                    |
C     |                                                      |
C     +------------------------------------------------------+
      SUBROUTINE ECHO4(K)
C
      IF (K .EQ. 1) GO TO 10
         WRITE(6,100)
         GO TO 11
   10 WRITE(6,101)
   11 CONTINUE
C
      RETURN
  100 FORMAT(//,' CONDUCTING BARRIER SEARCH.')
  101 FORMAT(//,' CONDUCTING AREA SEARCH.')
```

```
      END
C
C
C    ********************************************************************
C    ********************************************************************
C
C
C    +------------------------------------------------------------------+
C    |                                                                  |
C    |                       SUBROUTINE ECHO5                           |
C    |                                                                  |
C    +------------------------------------------------------------------+
C
      SUBROUTINE  ECHO5(X,Y)
C
      CALL  SKIP2
      WRITE(6,100)X,Y
C
      RETURN
  100 FORMAT(5X,'ANG2=',F7.2,5X,'START=',F7.2)
      END
C
C
C    ********************************************************************
C    ********************************************************************
C
C
C    +------------------------------------------------------------------+
C    |                                                                  |
C    |                       SUBROUTINE ECHO6                           |
C    |                                                                  |
C    +------------------------------------------------------------------+
C
      SUBROUTINE  ECHO6(R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12,I1,I2,R13,
     *                 R14,R15,L1,L2,L3,L4,I3,R16,R17,R18)
C
      REAL  R14(3),R15(3),R16(50),R17(50),R18(50)
C
      LOGICAL L1,L2,L3,L4
C
      CALL  PAGE
      CALL  SKIP2
      IF (L1) GO TO 2001
         WRITE(6,110)
         GO  TO  2002
 2001 WRITE(6,111)
 2002 IF (L3) GO TO 2003
         WRITE(6,112)
```

```
         GO TO 2004
 2003 WRITE(6,113)
 2004 CONTINUE
      IF (L4) WRITE(6,115)
      WRITE(6,102)
      WRITE(6,100)R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12,I1,I2,R13
C
      DO 10  I=1,3
         WRITE(6,101)I,R15(I),I,R14(I)
   10 CONTINUE
      IF (R14(2)  .EQ. 0.0) GO TO 2007
      G1=(R14(1)/R14(2))**2
      G2=(R14(3)/R14(2))**2
      RHO=SQRT((G1+1.0)*(G2+1.0))
      RHO=1.0/RHO
      WRITE(6,116) RHO
 2007 IF (L4) GO TO 2005
         RETURN
 2005 DO 2006 I=1,I3
         WRITE(6,114) I,R16(I),I,R17(I),I,R18(I)
 2006 CONTINUE
      RETURN
  100 FORMAT(5X,'FOMOD=',F7.2,5X,'FCMOS=',F7.2,5X,'SOC=',F7.2,5X,
     *         'SOS=',F7.2,/,5X,'TD=',F7.2,5X,'TS=',F7.2,5X,'FOMTC=',
     *          F7.2,5X,'FCMTS=',F7.2,/,5X,'STMIN=',F7.2,5X,'STMAX=',
     *          F7.2,5X,'RTSC=',F7.2,5X,'RTCC=',F7.2,/,5X,'SEED=',I10,
     *          5X,'NREP=',I4,5X,'TMAX=',F7.2)
  102 FORMAT(//,' REMAINING PLATFORM AND RUN PARAMETERS:')
  101 FORMAT(5X,'LAMBDA(',I1,')=',F7.2,5X,'SIGMA(',I1,')=',F7.2)
  110 FORMAT(/,' ACOUSTIC FLUCTUATION MODEL: LAMBDA-SIGMA JUMP.',/)
  111 FORMAT(/,' ACOUSTIC FLUCTUATION MODEL:  GAUSS-MARKOV.',/)
  112 FORMAT(/,' SIGNAL  INTEGRATION MODEL: NONE ( SE=>0, DETECTION )',/)
  113 FORMAT(/,' SIGNAL  INTEGRATION MODEL: MSEL ( 3 OF LAST 5 SE=>0, DET
     *ECTION)',/)
  114 FORMAT(,5X,'BRG(',I2,')=',F6.1,5X,'FOMBS(',I2,')=',F7.2,5X,'FCMBC(
     *',I2,')=',F7.2)
  115 FORMAT(/,' SEARCHER FIGURE OF MERIT DEPENDENT ON TARGET ASPECT.')
  116 FORMAT(10X,'COMPOUND ERROR FUNCTION CORRELATION = ',F7.4,/)
      END
C
C
C     ***************************************************************
C     ***************************************************************
C
C
C     +-------------------------------------------------------------+
C     |                      SUBROUTINE EXPO                        |
```

```
C     |                                                                    |
C     +--------------------------------------------------------------------+
C
      SUBROUTINE EXPC(ALAM,T,STACK)
C
      INTEGER SEED,C1,C2,C3
C
      REAL  STACK(5000)
C
      COMMON SEED,C1,C2,C3,C4
C
      C1=C1+1
      IF (C1 .LT. 5001) GO TO 10
         CALL LEXFN(SEED,STACK,5000,1,0)
         C1=1
   10 CONTINUE
      T=STACK(C1)*(1.0/ALAM)
      RETURN
      END
C
C
C     **********************************************************************
C     **********************************************************************
C
C
C     +--------------------------------------------------------------------+
C     |                                                                    |
C     |                        SUBROUTINE XLS                              |
C     |                                                                    |
C     +--------------------------------------------------------------------+
C
      SUBROUTINE XLS(SIGMA,X,STACK)
C
      INTEGER SEED,C1,C2,C3
C
      REAL  STACK(5000)
C
      COMMON SEED,C1,C2,C3,C4
C
      C2=C2+1
      IF (C2 .LT. 5001) GO TO 10
         CALL LNCFM(SEED,STACK,5000,1,0)
         C2=1
   10 CONTINUE
      X=SIGMA*STACK(C2)
      RETURN
      END
C
```

```
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                       SUBROUTINE UZ1                           |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE UZ1(U,STACK)
C
      INTEGER SEED,C1,C2,C3
C
      REAL STACK(5000)
C
      COMMON SEED,C1,C2,C3,C4
C
      C3=C3+1
      IF (C3 .LT. 5001) GO TO 10
         CALL LRND(SEED,STACK,5000,1,0)
         C3=1
   10 CONTINUE
      U=STACK(C3)
      RETURN
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                     SUBROUTINE SINKEM                          |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE SINKEM(NCO,NTDO,RNTCO,NCT,NTCT,RNTCT,NONE,NREP,
     *NCZDO,NCZDT,NCZO,NCZT,NBOTH,XCDT,YODT,XTOD,YTCD,XTDC,YTDC,
     *XCTD,YCTD,XMAX,YMAX,STATS1,STATS2,KBAR,FC,TC,FT,TT,M1,M2,AFLLS)
C
      REAL NTDO(NCO),RNTCO(NCO),NTDT(NDT),RNTCT(NCT),NCZDO(5),NCZDT(5),
     &F(5000),X(5000),XCDT(NCO),YCDT(NDC),XTOC(NCO),YTOD(NCO),XTDO(NDT),
     &YTCO(NCT),XCTD(NDT),YOTD(NDT),XL(4),YL(4),FC(M1),TO(M1),FT(M2),
     &TT(M2)
C
```

```
      LOGICAL STATS1,STATS2,AFLLS
C
      DCZO=0.0
      DCZT=0.0
      DTCT=FLOAT(NREP-NONE-NBOTH)
      NUM=NDO-(3*NDT)-(2*NBOTH)-NONE
      SMOE=(FLOAT(NUM)/FLOAT(4*NREP))+0.75
      NUM=NDT-(3*NDO)-(2*NBOTH)-NONE
      IF (KBAR .EQ. 2) NUM=NUM+(2.0*NONE)
      TMOE=(FLOAT(NUM)/FLOAT(4*NREP))+0.75
      IF (TMOE .LT. 0.00001) GO TO 2
      RATMOE=SMOE/TMOE
    2 CONTINUE
C
      IF (NCZO .LE. 0 .OR. NDO .LE. 0) GO TO 88
      DO 10 I=1,NCZO
         DCZO=DCZO+NCZDO(I)
         NCZDO(I)=NCZDO(I)/FLOAT(NDO)
   10 CONTINUE
      GO TO 89
   88 DCZO=0.0
   89 IF (NCZT .LE. 0 .OR. NDT .LE. 0) GO TO 77
      DO 9 I=1,NCZT
         DCZT=DCZT+NCZDT(I)
         NCZDT(I)=NCZDT(I)/FLOAT(NDT)
    9 CONTINUE
      GO TO 78
   77 DCZT=0.0
   78 IF (NCZO .LE. 0) DCZO=0.0
      IF (NDO .LE. 0) GO TO 79
      DCZO=DCZO/FLOAT(NDO)
      DCPO=1.0-DCZO
      GO TO 790
   79 DCPO=0.0
  790 IF (NDT .LE. 0) GO TO 791
      DCZT=DCZT/FLOAT(NDT)
      DCPT=1.0-DCZT
      GO TO 7911
  791 DCPT=0.0
 7911 PO=FLOAT(NDO)/FLOAT(NREP)
      POO=FLOAT(NDT)/FLOAT(NREP)
      IF(DCZO .GT. 0) GO TO 792
      DO 793 I=1,NCZO
         NCZDO(I)=0.0
  793 CONTINUE
  792 IF (DCZT .GT. 0) GO TO 794
      DO 795 I=1,NCZT
         NCZDO(I)=0.0
```

```
  795 CONTINUE
  794 CONTINUE
C
      WRITE(6,202)
      WRITE(8,202)
      WRITE(6,203) PD,DDPO,DCZO
      WRITE(8,203) PD,DDPO,DCZO
      IF (NCZO .LE. 0) GO TO 300
      DO 301 I=1,NCZO
         WRITE(6,204) I,NCZDO(I)
         WRITE(8,204) I,NCZDO(I)
  301 CONTINUE
  300 CONTINUE
      WRITE(6,207) SMOE
      WRITE(8,207) SMOE
      WRITE(6,205) PCD,CDPT,DCZT
      WRITE(8,205) PCD,CDPT,DCZT
      IF (NCZT .LE. 0) GO TO 302
      DO 303 I=1,NCZT
         WRITE(6,206) I,NCZDT(I)
         WRITE(8,206) I,NCZDT(I)
  303 CONTINUE
  302 CONTINUE
      WRITE(6,208) TMOE
      WRITE(8,208) TMOE
      IF (TMOE .LT. 0.00001) GO TO 3
      WRITE(6,209) RATMOE
      WRITE(8,209) RATMOE
    3 CONTINUE
C
      NPTSO=NDO+1
      IF (.NOT. STATS1) GO TO 30
      IF (NDO .LT. 10) GO TO 30
      IF (.NOT. STATS2) GO TO 31
      IF (NDO .LT. 500) GO TO 31
      KS=MINO(10,NDO/100)
      CALL SECTN(NTDO,NDO,KS)
      WRITE(6,113)
   31 CONTINUE
      CALL NOFIXF
      CALL HISTGF(NTDO,NDO,0)
      WRITE(6,100)
      DEL=1.0/FLOAT(NDO)
      F(1)=0.0
      X(1)=0.0
      WRITE(10,*) NPTSO
      DO 20 I=2,NPTSO
         F(I)=F(I-1)+DEL
```

```
         X(I)=NTDO(I-1)
   20 CONTINUE
      DO 21 I=1,NPTSO
         F(I)=F(I)*PD
   21 CONTINUE
         WRITE(10,210)(X(I),F(I), I=1,NPTSO)
      GO TO 40
   30 WRITE(10,*) NPTSO
      CALL PAGE
      IF (.NOT. STATS1 .AND. NDO .NE. 0) GO TO 40
      WRITE(6,102)
      IF (NDO .EQ. 0) GO TO 36
      DO 35 I=1,NDO
         WRITE(6,103)I,NTDO(I),I,RNTDO(I)
   35 CONTINUE
      GO TO 40
   36 WRITE(6,112)
      WRITE(8,112)
C
   40 CONTINUE
C
      NPTST=NDT+1
      IF (.NOT. STATS1) GO TO 50
      IF (NDT .LT. 10) GO TO 50
      IF (.NOT. STATS2) GO TO 51
      IF (NDT .LT. 500) GO TO 51
      KS=MINO(10,NDT/100)
      CALL SECTN(NTDT,NDT,KS)
      WRITE(6,115)
   51 CONTINUE
      CALL HISTGP(NTDT,NDT,0)
      WRITE(6,104)
      DEL=1.0/FLOAT(NDT)
      F(1)=0.0
      X(1)=0.0
      WRITE(10,*) NPTST
      DO 60 I=2,NPTST
         F(I)=F(I-1)+DEL
         X(I)=NTDT(I-1)
   60 CONTINUE
      DO 61 I=1,NPTST
         F(I)=F(I)*PCD
   61 CONTINUE
         WRITE(10,210)( X(I),F(I), I=1,NPTST)
      GO TO 70
   50 WRITE(10,*) NPTST
      CALL PAGE
      IF (.NOT. STATS1 .AND. NDT .NE. 0) GO TO 70
```

```
      WRITE(6,106)
      IF (NDT .EQ. 0) GO TO 56
      DO 55 I=1,NDT
         WRITE(6,107)I,NTDT(I),I,RNTDT(I)
   55 CONTINUE
      GO TO 70
   56 WRITE(6,111)
      WRITE(8,111)
C
   70 CONTINUE
      IF (.NOT. STATS1) GO TO 80
      IF (NDO .LT. 10) GO TO 80
      IF (.NOT. STATS2) GO TO 72
      IF (NDO .LT. 500) GO TO 72
      KS=MINO(10,NDO/100)
      CALL SECTN(RNTDO,NDO,KS)
      WRITE(6,116)
   72 CONTINUE
      CALL HISTGP(RNTDO,NDO,0)
      WRITE(6,108)
      DEL=1.0/FLOAT(NDO)
      F(1)=0.0
      X(1)=0.0
      DO 71 I=2,NPTSO
         F(I)=F(I-1)+DEL
         X(I)=RNTDO(I-1)
   71 CONTINUE
      DO 711 I=1,NPTSO
         F(I)=1.0-F(I)
  711 CONTINUE
         WRITE(10,210)(X(I),F(I),I=1,NPTSO)
         WRITE(10,210)(XODT(I),YODT(I),I=1,NDO)
         WRITE(10,210)(XTOD(I),YTOD(I),I=1,NDO)
C
   80 CONTINUE
      IF (.NOT. STATS1) GO TO 99
      IF (NDT .LT. 10) GO TO 99
      IF (.NOT. STATS2) GO TO 82
      IF (NDT .LT. 500) GO TO 82
      KS=MINO(10,NDT/100)
      CALL SECTN(RNTDT,NDT,KS)
      WRITE(6,117)
   82 CONTINUE
      CALL HISTGP(RNTDT,NDT,0)
      WRITE(6,110)
      DEL=1.0/FLOAT(NDT)
      F(1)=0.0
      X(1)=0.0
```

```
      DO 81 I=2,NPTST
         F(I)=F(I-1)+DEL
         X(I)=RNTCT(I-1)
   81 CONTINUE
      DO 811 I=1,NPTST
         F(I)=1.0-F(I)
  811 CONTINUE
         WRITE(10,210)(X(I),F(I),I=1,NPTST)
         WRITE(10,210)(XTOO(I),YTOO(I),I=1,NDT)
         WRITE(10,210)(XOTD(I),YOTD(I),I=1,NDT)
C
   99 CONTINUE
      IF (AFLLS) M1=-M1
      WRITE(10,*) M1
      IF (AFLLS) M1=-M1
      WRITE(10,210) (FO(I),TO(I),I=1,M1)
      WRITE(10,*) M2
      WRITE(10,210) (FT(I),TT(I),I=1,M2)
      RETURN
  100 FORMAT(/,5X,'DISTRIBUTION OF TIME TO SECURE DETECTION BY SEARCHER
     *')
  102 FORMAT(10X,'NUMBER OF SECURE DETECTIONS BY OWN SHIP LESS THAN TEN
     *(NO STATISTICS) -- DATA AS FOLLOWS:')
  103 FORMAT(15X,'NTOO(',I2,')=',F9.4,5X,'RNTOO(',I2,')=',F9.4)
  104 FORMAT(/,5X,'DISTRIBUTION OF TIME TO SECURE DETECTION BY TARGET')
  106 FORMAT(10X,'NUMBER OF SECURE DETECTIONS BY TARGET LESS THAN TEN (N
     *O STATISTICS) -- DATA AS FOLLOWS:')
  107 FORMAT(15X,'NTOT(',I2,')=',F9.4,5X,'RNTOT(',I2,')=',F9.4)
  108 FORMAT(/,5X,'DISTRIBUTION OF RANGE OF SECURE DETECTION BY SEARCHER
     *')
  110 FORMAT(/,5X,'DISTRIBUTION OF RANGE OF SECURE DETECTION BY TARGET')
  111 FORMAT(15X,'NO SECURE DETECTIONS MADE BY TARGET')
  112 FORMAT(15X,'NO SECURE DETECTIONS MADE BY OWN SHIP')
  113 FORMAT(/,5X,'RESULTS OF SECTIONING TIMES TO SECURE DETECTION BY SE
     *ARCHER')
  115 FORMAT(/,5X,'RESULTS OF SECTIONING TIMES TO SECURE DETECTION BY TA
     *GET')
  116 FORMAT(/,5X,'RESULTS OF SECTIONING RANGES OF SECURE DETECTION BY S
     *EARCHER')
  117 FORMAT(/,5X,'RESULTS OF SECTIONING RANGES OF SECURE DETECTION BY T
     *ARGET')
  202 FORMAT('1',//,5X,'SIMULATION RESULTS QUICK-SUMMARY:',/)
  203 FORMAT(10X,'PD(UNCONDITIONAL)= ',F7.4,/,10X,'PD(DIRECT-PATH,CONDIT
     *IONAL)= ',F7.4,/,10X,'PD(TOTAL-CZS,CONDITIONAL)= ',F7.4)
  204 FORMAT(10X,'PD(CZ-',I1,',CONDITIONAL)= ',F7.4)
  205 FORMAT(//,10X,'PCD(UNCONDITIONAL)= ',F7.4,/,10X,'PCD(DIRECT-PATH,C
     *ONDITIONAL)= ',F7.4,/,10X,'PCD(TOTAL-CZS,CONDITIONAL)= ',F7.4)
  206 FORMAT(10X,'PCD(CZ-',I1,',CONDITIONAL)= ',F7.4)
```

```
  207 FORMAT(10X,'MOE(SEARCHER)= ',F7.4)
  208 FORMAT(10X,'MOE(TARGET)= ',F7.4)
  209 FORMAT(/,10X,'EXCHANGE RATIO= ',E12.4)
  210 FORMAT(8F9.3)
      END
C
C
C    *******************************************************************
C    *******************************************************************
C
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |                       SUBROUTINE PAGE                           |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
      SUBROUTINE PAGE
      WRITE(6,100)
      RETURN
  100 FORMAT('1')
      END
C
C
C    *******************************************************************
C    *******************************************************************
C
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |                       SUBROUTINE SKIP2                          |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
      SUBROUTINE SKIP2
         WRITE(6,100)
         WRITE(8,100)
         RETURN
  100 FORMAT(//)
      END
C
C
C    *******************************************************************
C    *******************************************************************
C
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
```

```
C     |                          SUBROUTINE CLEAR                          |
C     |                                                                    |
C     +--------------------------------------------------------------------+
C
      SUBROUTINE CLEAR
         CALL FRTCMS ('CLRSCRN ')
      RETURN
      END
C
C
C     **********************************************************************
C     **********************************************************************
C
C
C
C     +--------------------------------------------------------------------+
C     |                                                                    |
C     |                          SUBROUTINE DESCRB                         |
C     |                                                                    |
C     +--------------------------------------------------------------------+
C
      SUBROUTINE DESCRB
C
      WRITE(8,100)
      WRITE(8,101)
      WRITE(8,102)
      WRITE(8,103)
   11 WRITE(8,104)
      READ(8,*) GONOGO
      IF (GONOGO .NE. 1.0 .AND. GONOGO .NE. 2.0) GO TO 10
      IF (GONOGO .EQ. 1.0) GO TO 99
      RETURN
   10 WRITE(8,105)
      GO TO 11
   99 CONTINUE
      STOP
C
  100 FORMAT(//,'     PROGRAM PASS IS A MONTE-CARLO SIMULATION OF PASSIV
     *E ACOUSTIC',/,'SEARCH WHICH EMPLOYS A THRESHOLD-CROSSING MODEL TO
     *DETERMINE IF DETECTION HAS',/,'OCCURRED, AND EITHER A LAMBDA-SIGMA
     * JUMP OR GAUSS MARKOV STOCHASTIC ERROR',/,'PROCESS TO MODEL ACOUST
     *IC FLUCTUATIONS.')
  101 FORMAT(/,'     DURING THE INTERACTIVE STAGE OF THE PROGRAM AN INPU
     *T DATA FILE WILL BE',/,'CONSTRUCTED ON YOUR A-DISK (PASS DATA). YO
     *U MAY KEEP THIS FILE FOR FUTURE RUNS,',/,'OR ERASE IT AND CONSTRUC
     *T ANOTHER THE NEXT TIME YOU RUN PASS.')
  102 FORMAT(/,'     ALL OUTPUT FROM THE PROGRAM WILL BE SENT TO YOUR A-
     *DISK (PASS OUTPUT).',/,'SOME SELECTED OUTPUT WILL GO DIRECTLY TO T
     *HE TERMINAL SCREEN. DEPENDING',/,'ON THE OUTPUT OPTION YOU SELECT,
```

```
     * THE OUTPUT FILE MAY EXCEED THE ALLOWABLE',/,'RECORD LENGTH FOR A
     *NORMAL PRINT COMMAND. FOR THIS REASON, ALWAYS USE',/,'THE COMMAND
     *"PRINT PASS OUTPUT (CC" WHEN REQUESTING HARD-COPY.')
  103 FORMAT(/,'     WHILE SOME ERROR CHECKING IS DONE DURING THE INTERA
     *CTIVE DATA INPUT',/,'PORTION OF THE PROGRAM, IT IS NOT IMPOSSIBLE
     *TO INPUT BAD DATA. IF THE PRO-',/,'GRAM BOMBS, IT IS MOST LIKELY D
     *UE TO INVALID, OR INCONSISTENT DATA.',////)
  104 FORMAT(/,' OPTION NO. 1: TERMINATE OR PROCEED.',/,5X,'SELECT ONE O
     *F THE FOLLOWING:',/,10X,'1. TERMINATE PROGRAM.',/,10X,'2. PROCEED
     *WITH PASS.',/,5X,'ENTER 1 OR 2.',/,5X,'NOTE: IF YOU TERMINATE NOW,
     *THE DATA IN FILE PASS DATA WILL',/,11X,'BE IN FILE PASS HOLD, AND
     *FILE PASS DATA WILL BE EMPTY.')
  105 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                       SUBROUTINE OPTN2A                        |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN2A(INTEG)
C
      LOGICAL INTEG
C
      CALL CLEAR
   11 WRITE(8,100)
      READ(8,*) REPLY
      IF (REPLY .NE. 1.0 .AND. REPLY .NE. 2.0) GO TO 10
      IF (REPLY .EQ. 1.0) GO TO 12
         INTEG=.TRUE.
         GO TO 13
   12 INTEG=.FALSE.
      GO TO 13
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   13 CONTINUE
      RETURN
C
  100 FORMAT(//,' OPTION NO. 2A: SELECT SIGNAL INTEGRATION MODEL.',/,5X,
     *'SELECT ONE OF THE FOLLOWING:',/,10X,'1. NO SIGNAL INTEGRATION (SE
     *=>0, DETECTION ).',/,10X,'2. MSEL SIGNAL INTEGRATION ( 3 OF LAST 5
```

```
     * SE=>0, DETECTION).',/,5X,'NOTE: MSEL SIGNAL INTEGRATION WILL SIGN
     *IFICANTLY INCREASE ',/,11X,'PROGRAM EXECUTION TIME IN SOME SCENARI
     *OS.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                        SUBROUTINE OPTN2                        |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN2(AFLLS,AFLGM)
C
      LOGICAL AFLLS,AFLGM
C
      AFLLS=.FALSE.
      AFLGM=.FALSE.
C
      CALL CLEAR
   11 WRITE(6,100)
      READ(8,*) FLUCT
C
      IF (FLUCT .NE. 1.0 .AND. FLUCT .NE. 2.0) GO TO 10
      IF (FLUCT .EQ. 1.0) GO TO 12
         AFLGM=.TRUE.
         GO TO 13
   12 AFLLS=.TRUE.
      GO TO 13
   10 CALL CLEAR
      WRITE(6,101)
      GO TO 11
   13 CONTINUE
      RETURN
  100 FORMAT(//,' OPTION NO. 2: SELECT ACOUSTIC FLUCTUATION MODEL.',/,5X
     *,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. LAMBDA-SIGMA JUMP.',/,10
     *X,'2. GAUSS-MARKOV.',/,5X,'ENTER 1 OR 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
      END
C
C
C     ******************************************************************
C     ******************************************************************
```

```
C
C
C     +------------------------------------------------------------------+
C     |                                                                  |
C     |                       SUBROUTINE OPTN3                           |
C     |                                                                  |
C     +------------------------------------------------------------------+
C
C
      SUBROUTINE OPTN3(XMAX,YMAX,FILE)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF(FILE) GO TO 9
         XMAX=0.0
         YMAX=0.0
         GO TO 11
    9 READ(9,*) XMAX,YMAX
   11 WRITE(8,100)
      READ(8,*) AREA
      J=INT(AREA)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) XMAX,YMAX
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) XMAX,YMAX
      WRITE(10,*) XMAX,YMAX
      RETURN
   40 CALL CLEAR
      WRITE(8,103)
      READ(8,*) XMAX,YMAX
      WRITE(8,102) XMAX,YMAX
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO. 3: SELECT AREA SIZE.',/,5X,'SELECT ONE OF T
     *HE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA (IN FILE PASS DATA).
     *',/,10X,'2. ACCEPT EXISTING AREA DIMENSIONS.',/,10X,'3. CHANGE ARE
     *A DIMENSIONS.',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OPTION
     * YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(//,'XMAX= ',F9.4,5X,'YMAX= ',F9.4,/,5X,'ENTER 0 TO RETURN T
```

```
     *C THE AREA SIZE MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (X
     *MAX,YMAX).')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                      SUBROJTINE OPTN3A                         |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN3A(FILE)
C
      LOGICAL FILE
C
      INTEGER SEED
C
      CALL CLEAR
   11 WRITE(8,100)
      READ(8,*) FFILE
      J=INT(FFILE)
      IF (J .NE. 1 .AND. J .NE. 2) GO TO 10
      IF (J .EQ. 1) GO TO 12
         FILE=.TRUE.
         RETURN
   12 FILE=.FALSE.
      RETURN
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
C
  100 FORMAT(//,' OPTION 3A: EXISTENCE OF FILE PASS DATA.',5X,'SELECT ON
     *E OF THE FOLLOWING:',/,10X,'1. FILE PASS DATA IS NOT ON A-DISK.',/
     *,10X,'2. FILE PASS DATA IS ON A-DISK.',/,5X,'SELECT 1 OR 2.')
  101 FORMAT(/,' ***** INCORRECT RESPUNSE, TRY AGAIN    *****')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
```

```
C     |                                                                |
C     |                       SUBROUTINE OPTN4                         |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN4(NP,KP,XP,YP,FILE)
C
      REAL XP(50),YP(50)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         NP=2
         KP=1
         XP(1)=0.0
         XP(2)=0.0
         YP(1)=0.0
         YP(2)=0.0
         GO TO 11
    9 READ(9,*) NP,KP
      DO 10 I=1,NP
         READ(9,*) XP(I),YP(I)
   10 CONTINUE
   11 WRITE(8,100)
      READ(8,*) TRACK
      J=INT(TRACK)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 12
      GO TO (20,30,40), J
   20 CALL CLEAR
      WRITE(8,102) NP,KP
      DO 21 I=1,NP
         WRITE(8,103) I,XP(I),I,YP(I)
   21 CONTINUE
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) NP,KP
      WRITE(10,*) NP,KP
      DO 31 I=1,NP
         WRITE(7,*) XP(I),YP(I)
         WRITE(10,*) XP(I),YP(I)
   31 CONTINUE
      RETURN
   40 CALL CLEAR
      WRITE(8,105)
      READ(8,*) NP,KP
```

```
      DO 41 I=1,NF
         CALL CLEAR
         WRITE(8,102) NF,KF
         II=I-1
         IF(I .EQ. 1) GO TO 42
         DO 42 J=1,II
            WRITE(8,103) J,XP(J),J,YP(J)
 42      CONTINUE
         WRITE(8,106) I
         READ(8,*) XP(I),YP(I)
 41   CONTINUE
      GO TO 20
 12   WRITE(8,101)
      CALL CLEAR
      GO TO 11
100   FORMAT(//,' OPTION NO. 4: SPECIFY SEARCHER TRACK.',/,5X,'SELECT ON
     *E OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,'2. AC
     *CEPT EXISTING SEARCHER TRACK.',/,10X,'3. CHANGE SEARCHER TRACK.',/
     *,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY CHOOSE NO. 2.')
101   FORMAT(/, '***** INCORRECT RESPONSE, TRY AGAIN    *****')
102   FORMAT(/,10X,'NP=  ',I2,5X,'KP=  ',I2)
103   FORMAT(10X,'XP(',I2,')=',F9.4,5X,'YP(',I2,')=',F9.4)
104   FORMAT(/, 'ENTER 0 TO RETURN TO SEARCHER TRACK MENU.')
105   FORMAT(/, 'ENTER THE NUMBER OF ANCHOR POINTS AND RETURN POINT AS A
     * DATA PAIR (NP,KP):')
106   FORMAT(/, 'ENTER ANCHOR POINT NO. ',I2,' AS A DATA PAIR (XP,YP)')
      END
C
C
C    ******************************************************************
C    ******************************************************************
C
C
C    +----------------------------------------------------------------+
C    |                                                                |
C    |                     SUBROUTINE OPTN5                           |
C    |                                                                |
C    +----------------------------------------------------------------+
C
      SUBROUTINE OPTN5(MC,RC,CL,FILE)
C
      REAL RC(20),CL(20)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         MO=1
```

```
      RO(1)=0.C
      OL(1)=0.C
      GO TO 11
    9 READ(9,*) MC,(RO(I),CL(I), I=1,MO)
   11 WRITE(8,10C)
      READ(8,*) CSPL
      J=INT(CSPL)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GC TO 12
      GC TO (20,3C,40),J
   2C CALL CLEAR
      WRITE(8,102) MC
      DC 21 I=1,MC
         WRITE(8,103) I,RC(I),I,OL(I)
   21 CCNTINUE
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GC TO 11
   3C CALL CLEAR
      WRITE(7,*) MC
      WRITE(10,*) MO
      CC 31 I=1,MC
         WRITE(7,*) RO(I),CL(I)
         WRITE(1C,*) RO(I),OL(I)
   31 CCNTINUE
      RETURN
   4C CALL CLEAR
      WRITE(8,105)
      READ(8,*) MC
      CC 41 I=1,MC
         CALL CLEAR
         WRITE(8,102) MC
         II=I-1
         IF(I .EC. 1) GC TC 42
         DO 42 J=1,II
            WRITE(8,103) J,RO(J),J,OL(J)
   42    CONTINUE
         WRITE(8,106) I
         READ(8,*) RC(I),CL(I)
   41 CCNTINUE
      GC TO 20
   12 WRITE(8,101)
      GC TO 11
C
  10C FCRMAT(//,'CPTICN NC. 5: SEARCHER PROPAGATICN LCSS DATA.',/,5X,'SE
     *LECT CNE CF THE FCLLCWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X
     *,'2. ACCEPT EXISTING PRCP-LOSS DATA.',/,10X,'3. CHANGE PRCP-LCSS D
     *ATA.',/,5X,'TO EXIT THIS OPTICN YCU MUST EVENTUALLY CHOCSE NC. 2.'
```

```
     *)
 1C1 FCFMAT(/,' ***** INCCRRECT RESPONSE, TRY AGAIN     *****')
 1C2 FCFMAT(/,1CX,'MO= ',I2)
 1C3 FCRMAT(10X,'RO(',I2,')=',F9.4,5X,'CL(',I2,')=',F9.4)
 1C4 FCRMAT(/,' ENTER 0 TC RETURN TO SEARCHER PRCF-LCSS MENU.')
 1C5 FCFMAT(/,' ENTER THE NUMBER OF RANGE, PROP-LCSS DATA PAIRS (MC):')
 1C6 FCRMAT(/,' ENTER RANGE (NM) AND PROP-LOSS (CB/UPA/YC) AS A DATA',/
     *,' PAIR (RC,OL) FCR CATA POINT NO. ',I2,' :')
      END
C
C
C    ********************************************************************
C    ********************************************************************
C
C
C    +------------------------------------------------------------------+
C    |                                                                  |
C    |                         SUBROUTINE OPTN6                         |
C    |                                                                  |
C    +------------------------------------------------------------------+
C
      SUBROUTINE CPTN6(MT,RT,TL,FILE)
C
      REAL RT(20),TL(20)
C
      LCGICAL FILE
C
      CALL CLEAR
      IF (FILE) GC TO 9
         MT=1
         RT(1)=0.C
         TL(1)=0.C
         GO TO 11
    9 READ(9,*) MT,(RT(I),TL(I), I=1,MT)
   11 WRITE(8,10C)
      READ(8,*) TSPL
      J=INT(TSPL)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GC TC 12
      GC TO (20,3C,40), J
   2C CALL CLEAR
      WRITE(8,102) MT
      CC 21 I=1,MT
         WRITE(8,103) I,RT(I),I,TL(I)
   21 CCNTINUE
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GC TO 11
```

```
 20 CALL CLEAR
    WRITE(7,*) MT
    WRITE(10,*) MT
    DO 31  I=1,MT
       WRITE(7,*) RT(I),TL(I)
       WRITE(10,*) RT(I),TL(I)
 31 CONTINUE
    RETURN
 40 CALL CLEAR
    WRITE(8,105)
    READ(8,*) MT
    DO 41  I=1,MT
       CALL CLEAR
       WRITE(8,102) MT
       II=I-1
       IF  (I  .EQ. 1)  GO TO 42
       DO  42  J=1,II
          WRITE(8,103 ) J,RT(J),J,TL(J)
 42    CONTINUE
       WRITE(8,106) I
       READ(8,*) RT(I),TL(I)
 41 CONTINUE
    GO TO 20
 12 WRITE(8,101)
    GO TO 11
C
100 FORMAT(//,'OPTION NO. 6: TARGET PROPAGATION LOSS DATA.',/,5X,'SELE
   *CT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,'
   *2. ACCEPT EXISTING PROP-LOSS DATA.',/,10X,'3. CHANGE PROP-LOSS DAT
   *A.',/,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY CHOOSE NO. 2.')
101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
102 FORMAT(/,10X,'MT= ',I2)
103 FORMAT(10X,'RT(',I2,')=',F9.4,5X,'TL(',I2,')=',F9.4)
104 FORMAT(/,' ENTER 0 TO RETURN TO TARGET PROP-LOSS MENU.')
105 FORMAT(/,' ENTER THE NUMBER OF RANGE, PROP-LOSS DATA PAIRS (MT):')
106 FORMAT(/,' ENTER RANGE (NM) AND PROP-LOSS (DB/UFA/YD) AS A DATA',/
   *,' PAIR (RT,TL) FOR DATA POINT NO. ',I2,' :')
    END
C
C
C    **********************************************************************
C    **********************************************************************
C
C
C    +--------------------------------------------------------------------+
C    |                                                                    |
C    |                       SUBROUTINE OPTN7                             |
C    |                                                                    |
```

```
C     +-----------------------------------------------------------------+
C
      SUBROUTINE CPTN7(NCZO,RCZO,CZLO,FILE)
C
      REAL RCZO(5,2),CZLO(5)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         NCZO=1
         RCZO(1,1)=0.0
         RCZO(1,2)=0.0
         CZLO(1)=0.0
         GO TO 11
    9 READ(9,*) NCZO
      IF (NCZO .GT. 0) READ(9,*) ((RCZO(I,J),J=1,2),CZLO(I),I=1,NCZO)
   11 WRITE(8,100)
      READ(8,*) OSCZ
      J=INT(OSCZ)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 12
      GO TO (20,30,40),J
   20 CALL CLEAR
      WRITE(8,102) NCZO
      IF (NCZO .EQ. 0) GO TO 21
      DO 21 I=1,NCZO
         WRITE(8,103) I,RCZO(I,1),I,RCZO(I,2),I,CZLO(I)
   21 CONTINUE
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 CALL CLEAR
      WRITE(7,*) NCZO
      WRITE(10,*) NCZO
      IF (NCZO .EQ. 0) GO TO 31
      DO 31 I=1,NCZO
         WRITE(7,*) RCZO(I,1),RCZO(I,2),CZLO(I)
         WRITE(10,*) RCZO(I,1),RCZO(I,2),CZLO(I)
   31 CONTINUE
      RETURN
   40 CALL CLEAR
      WRITE(8,105)
      READ(8,*) NCZO
      IF (NCZO .EQ.0) GO TO 20
      DO 41 I=1,NCZO
         CALL CLEAR
         WRITE(8,102) NCZO
```

```
         II=I-1
         IF (I .EQ. 1) GO TO 42
         DO 42 J=1,II
            WRITE(8,103) J,RCZO(J,1),J,RCZO(J,2),J,CZLO(J)
  42     CONTINUE
         WRITE(8,106) I
         READ(8,*) RCZO(I,1),RCZO(I,2),CZLO(I)
  41  CONTINUE
      GO TO 20
  12  WRITE(8,101)
      GO TO 11
C
 100  FORMAT(//,'OPTION NO. 7: SEARCHER CONVERGENCE ZONES.  ',/,5X,'SELE
     *CT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,'
     *2. ACCEPT EXISTING CONV-ZONE DATA.',/,10X,'3. CHANGE CONV-ZONE DAT
     *A.',/,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY CHOOSE NO. 2.')
 101  FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN    *****')
 102  FORMAT(/,10X,'NCZO= ',I1)
 103  FORMAT(10X,'RCZO(',I1,'1)= ',F9.4,5X,'RCZO(',I1,'2)= ',F9.4,5X,'CZ
     *LO(',I1,')= ',F9.4)
 104  FORMAT(/,' ENTER 0 TO RETURN TO SEARCHER CONV-ZONE MENU.')
 105  FORMAT(/,' ENTER NUMBER OF SEARCHER CONVERGENCE ZONES (NCZO):')
 106  FORMAT(/,' ENTER INNER-RANGE (NM), OUTER RANGE (NM), AND PROP-LOSS
     * (DB/UPA/YD)',/,' AS A DATA TRIPLET (RCZO(IN),RCZO(OUT),CZLO) FOR
     *DATA POINT NO. ',I1)
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                        SUBROUTINE OPTN8                        |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN8(NCZT,RCZT,CZLT,FILE)
C
      REAL RCZT(5,2),CZLT(5)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         NCZT=1
         RCZT(1,1)=0.0
```

```
      RCZT(1,2)=0.0
      CZLT(1)=0.0
      GO TO 11
    9 READ(9,*) NCZT
      IF (NCZT .GT. 0) READ(9,*) ((RCZT(I,J),J=1,2),CZLT(I),I=1,NCZT)
   11 WRITE(8,100)
      READ(8,*) IACZ
      J=INT(IACZ)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 12
      GO TO (20,30,40), J
   20 CALL CLEAR
      WRITE(8,102) NCZT
      IF (NCZT .EQ. 0) GO TO 21
      DO 21 I=1,NCZT
         WRITE(8,103) I,RCZT(I,1),I,RCZT(I,2),I,CZLT(I)
   21 CONTINUE
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 CALL CLEAR
      WRITE(7,*) NCZT
      WRITE(10,*) NCZT
      IF (NCZT .EQ. 0) GO TO 31
      DO 31 I=1,NCZT
         WRITE(7,*) RCZT(I,1),RCZT(I,2),CZLT(I)
         WRITE(10,*) RCZT(I,1),RCZT(I,2),CZLT(I)
   31 CONTINUE
      RETURN
   40 CALL CLEAR
      WRITE(8,105)
      READ(8,*) NCZT
      IF (NCZT .EQ. 0) GO TO 20
      DO 41 I=1,NCZT
         CALL CLEAR
         WRITE(8,102) NCZT
         II=I-1
         IF (I .EQ. 1) GO TO 42
         DO 42 J=1,II
            WRITE(8,103) J,RCZT(J,1),J,RCZT(J,2),J,CZLT(J)
   42    CONTINUE
         WRITE(8,106) I
         READ(8,*) RCZT(I,1),RCZT(I,2),CZLT(I)
   41 CONTINUE
      GO TO 20
   12 WRITE(8,101)
      GO TO 11
C
```

```
100 FORMAT(//,'OPTION NO. 8: TARGET CONVERGENCE ZONES.    ',/,5X,'SELE
   *CT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,'
   *2. ACCEPT EXISTING CONV-ZONE DATA.',/,10X,'3. CHANGE CONV-ZONE DAT
   *A.',/,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY CHOOSE NO. 2.')
101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN    *****')
102 FORMAT(/,10X,'NCZT= ',I1)
103 FORMAT(10X,'RCZT(',I1,'1)= ',F9.4,5X,'RCZT(',I1,'2)= ',F9.4,5X,'CZ
   *LT(',I1,')= ',F9.4)
104 FORMAT(/,' ENTER 0 TO RETURN TO TARGET CONV-ZONE MENU.')
105 FORMAT(/,' ENTER NUMBER OF TARGET CONVERGENCE ZONES (NCZT):')
106 FORMAT(/,' ENTER INNER-RANGE (NM), OUTER RANGE (NM), AND PROP-LOSS
   * (DB/UPA/YD)',/,' AS A DATA TRIPLET (RCZT(IN),RCZT(OUT),CZLT) FOR
   *DATA POINT NO. ',I1)
    END
C
C
C   ****************************************************************
C   ****************************************************************
C
C
C   +--------------------------------------------------------------+
C   |                                                              |
C   |                     SUBROUTINE OPTN9                         |
C   |                                                              |
C   +--------------------------------------------------------------+
C
    SUBROUTINE OPTN9(K,S,A,FILE,SPECX)
C
    LOGICAL FILE,SPECX
C
    CALL CLEAR
    IF (FILE) GO TO 9
       K=1
       S=0.0
       A=0.0
       GO TO 11
  9 READ(9,*) K
    IF (K .EQ. 2) READ(9,*) S,A
 11 WRITE(8,100)
    READ(8,*) EARR
    K=INT(EARR)
C
    IF (K .NE. 1 .AND. K .NE. 2) GO TO 10
       GO TO 13
 10 CALL CLEAR
    WRITE(8,101)
    GO TO 11
 13 CONTINUE
```

```
      IF (K  .EQ. 1) GO TO 14
         WRITE(8,102)
         READ(8,*) S,A
      CALL CLEAR
   16 WRITE(8,103)
      READ(8,*) REPLY
      IF (REPLY .NE. 1.0 .AND. REPLY .NE. 2.0) GO TO 15
      IF (REPLY .EQ. 1.0) GO TO 14
      SPECX=.TRUE.
      GO TO 14
   15 CALL CLEAR
      WRITE(8,101)
      GO TO 16
   14 WRITE(7,*) K
      IF (K  .EQ. 2) WRITE(7,*) S,A
      RETURN
  100 FORMAT(//,' OPTION NO. 9: SELECT AREA OR BARRIER SEARCH.     ',/,5X
     *,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. AREA SEARCH.      ',/,10
     *X,'2. BARRIER SEARCH.',/,5X,'ENTER 1 OR 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN    *****')
  102 FORMAT(/,' BARRIER SEARCH SELECTED. ENTER THE MAX SEARCHER TIME LA
     *TE (START) ',/,' AND MAX COURSE VARIATION (ANG2) AS A DATA PAIR:')
  103 FORMAT(//,' OPTION NO. 9A: TARGET X-DISTRIBUTION.',/,5X,'SELECT ON
     *E OF THE FOLLOWING:',/,10X,'1. TARGET INITIALLY UNIFORM ON (0,XMAX
     *).',/,10X,'2. USER SPECIFY TARGET DISTRIBUTION ON (0,XMAX).',/,5X,
     *'ENTER 1 OR 2.')
      END
C
C
C    *******************************************************************
C    *******************************************************************
C
C
C
C    +-----------------------------------------------------------------+
C    |                                                                 |
C    |                      SUBROUTINE OPTN9B                          |
C    |                                                                 |
C    +-----------------------------------------------------------------+
C
      SUBROUTINE OPTN9B(XMAX,NBINS,PBIN,XBIN)
C
      REAL PBIN(20),XBIN(20)
      CALL CLEAR
C
    5 WRITE(8,100)
      READ(8,*) NBINS
      IF (NBINS .LE. 1 .OR. NBINS .GT. 20) GO TO 20
C
```

```
      SUM=0.0
      DO 10 I=1,NBINS
         CALL CLEAR
         WRITE(8,102) I
         READ (8,*) XBIN(I),PBIN(I)
         SUM=SUM+PBIN(I)
   10 CONTINUE
      CALL CLEAR
C
      WRITE(8,103)
      DO 11 I=1,NBINS
         WRITE(8,104) I,XBIN(I),I,PBIN(I)
   11 CONTINUE
C
      IF (SUM .GT. 1.001 .OR. SUM .LT. 0.999) GO TO 21
      IF (XBIN(NBINS) .NE. XMAX) GO TO 22
      WRITE(8,107)
      READ(8,*) REPLY
      IF (REPLY .NE. 1.0) GO TO 5
      RETURN
C
   20 CALL CLEAR
      WRITE(8,101)
      GO TO 5
C
   21 WRITE(8,105)
      READ(8,*) REPLY
      CALL CLEAR
      GO TO 5
C
   22 WRITE(8,106)
      READ(8,*) REPLY
      CALL CLEAR
      GO TO 5
C
  100 FORMAT(//,' SPECIFY X-DISTRIBUTION OF TARGET IN BARRIER SCENARIO.'
     *,/,5X,'ENTER NUMBER OF PROBABILITY "BINS" ON (0,XMAX) (NBINS):')
  101 FORMAT(//,' **** ERROR: NBINS MUST BE BETWEEN 2 AND 20 ****')
  102 FORMAT(//,' ENTER RIGHT LIMIT (XBIN) AND PROBABILITY OF INCLUSION
     *(PBIN) AS',/,' A DATA PAIR (XBIN,PBIN) FOR DATA POINT ',I2)
  103 FORMAT(//,' DATA REVIEW:')
  104 FORMAT(5X,'XBIN(',I2,')=',F9.4,5X,'PBIN(',I2,')=',F7.4)
  105 FORMAT(//,' **** ERROR: SUM OF PBIN MUST EQUAL 1.00 ****',/,5X,'EN
     *TER 0 TO RE-ENTER DATA.')
  106 FORMAT(//,' **** ERROR: LAST XBIN VALUE MUST EQUAL XMAX ****',/,5X
     *,'ENTER 0 TO RE-ENTER DATA.')
  107 FORMAT(//,' ENTER: 1 TO ACCEPT DATA OR 0 TO RE-ENTER DATA.')
      END
```

```
C
C
C     ************************************************************
C     ************************************************************
C
C
C     +----------------------------------------------------------+
C     |                                                          |
C     |                   SUBROUTINE OPTN10                      |
C     |                                                          |
C     +----------------------------------------------------------+
C
      SUBROUTINE OPTN10(FOMCD,FOMOS,FILE,ASPECT)
C
      LOGICAL FILE,ASPECT
C
      CALL CLEAR
      IF (FILE) GO TO 9
         FOMCD=0.0
         FOMCS=0.0
         GO TO 12
    9 READ (9,*) FOMCD,FOMCS
   12 WRITE(6,104)
      READ(8,*) REPLY
      IF (REPLY .NE. 1.0 .AND. REPLY .NE. 2.0) GO TO 13
      IF (REPLY .EQ. 1.0) GO TO 11
         ASPECT=.TRUE.
         FOMCD=0.0
         FOMCS=0.0
         GO TO 30
   13 CALL CLEAR
      WRITE(6,101)
      GO TO 12
   11 WRITE(6,100)
      READ(8,*) FOMO
      J=INT(FOMO)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(6,101)
      GO TO 11
   20 WRITE(8,102) FOMCD,FOMCS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) FOMCD,FOMCS
      RETURN
   40 CALL CLEAR
```

```
      WRITE(8,103)
      READ(8,*) FOMOD,FOMCS
      WRITE(8,102) FOMOD,FOMCS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.10: SEARCHER FIGURE OF MERIT.',/,5X,'SELECT
     *ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,'2.
     *ACCEPT EXISTING FIGURE OF MERIT DATA.',/,10X,'3. CHANGE FIGURE OF
     *MERIT DATA.  ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OPTION
     * YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' *****  INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(//,'FOMOD= ',F9.4,5X,'FOMCS= ',F9.4,/,5X,'ENTER 0 TO RETURN
     *TO THE SEARCHER FIGURE OF MERIT MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (F
     *OMOD,FOMCS):')
  104 FORMAT(//,' OPTION NO. 10A: SEARCHER FOM DEPENDENT ON TARGET ASPEC
     *T?',/,5X,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. SEARCHER FOM NOT
     * TARGET ASPECT DEPENDENT.',/,10X,'2. SEARCHER FOM TARGET ASPECT DE
     *PENDENT.',/,5X,'ENTER 1 OR 2.')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                      SUBROUTINE OPT10B                         |
C     |                                                                |
C     +----------------------------------------------------------------+
C
C
      SUBROUTINE OPT10B(NLS,BRG,FOMBS,FOMBD)
C
      REAL BRG(50),FOMBS(50),FOMBD(50)
C
      CALL CLEAR
    5 WRITE(8,100)
      READ(8,*) NLS
      IF (NLS .LE. 3 .OR. NLS .GT. 50) GO TO 20
      DO 10 I=1,NLS
         CALL CLEAR
         WRITE(8,102) I
         READ(8,*) BRG(I),FOMBS(I),FOMBD(I)
   10 CONTINUE
```

```
C
      CALL CLEAR
      WRITE(8,103)
      DO 11 I=1,NLS
         WRITE(8,104) I,BRG(I),I,FOMBS(I),I,FOMBD(I)
   11 CONTINUE
C
      WRITE(8,107)
      READ(8,*) REPLY
      IF (REPLY .NE. 1.0) GO TO 5
      NLS=NLS+1
      BRG(NLS)=360.0
      FOMBS(NLS)=FOMBS(1)
      FOMBD(NLS)=FOMBD(1)
      RETURN
C
   20 CALL CLEAR
      WRITE(8,101)
      GO TO 5
C
  100 FORMAT(//,' OPTION NO. 10B: SEARCHER FOM DEPENDENT ON TARGET ASPEC
     *T.',/,5X,'ENTER NUMBER OF DATA POINTS (NLS):')
  101 FORMAT(/,' ***** ERROR: NLS MUST BE => 4 AND <= 50. *****')
  102 FORMAT(/,' ENTER DATA POINT ',I2,' AS A TRIPLET (BRG,FOMBS,FOMBD),
     * WHERE:',/,5X,'BRG=RELATIVE BEARING (ANGLE CW FROM TARGET BOW, IN
     *DEGREES)',/,5X,'FOMBS=SEARCHER SPRINT FIGURE-OF-MERIT AT BRG (DB)'
     *,/,5X,'FOMBD=SEARCHER DRIFT FIGURE-OF-MERIT AT BRG (DB)',/,' NOTE:
     * (1). ENTER DATA IN ORDER OF ASCENDING BEARING (0-360).',/,7X,'(2)
     *. THE FIRST POINT ENTERED MUST BE FOR BRG=0.0 (ON THE BOW).')
  103 FORMAT(//,' DATA REVIEW:')
  104 FORMAT(5X,'BRG(',I2,')=',F6.1,5X,'FOMBS(',I2,')=',F7.2,5X,'FOMBD('
     *,I2,')=',F7.2)
  107 FORMAT(//,' ENTER: 1 TO ACCEPT DATA OR 0 TO RE-ENTER DATA.')
      END
C
C
C     ****************************************************************
C     ****************************************************************
C
C
C     +--------------------------------------------------------------+
C     |                                                              |
C     |                      SUBROUTINE OPTN11                       |
C     |                                                              |
C     +--------------------------------------------------------------+
C
      SUBROUTINE OPTN11(SCD,SCS,FILE)
C
```

```
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         SOD=0.0
         SOS=0.0
         GO TO 11
    9 READ (9,*) SOD,SOS
   11 WRITE(8,100)
      READ(8,*) SOSO
      J=INT(SOSO)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) SOD,SOS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) SOD,SOS
      RETURN
   40 CALL CLEAR
      WRITE(8,103)
      READ(8,*) SOD,SOS
      WRITE(8,102) SOD,SOS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.11: SEARCHER SPRINT/DRIFT SPEEDS.',/,5X,'SEL
     *ECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,
     *'2. ACCEPT EXISTING SPRINT/DRIFT SPEEDS.',/,10X,'3. CHANGE SPRINT/
     *DRIFT SPEEDS.',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OPTION
     * YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(//,'SOD= ',F9.4,5X,'SOS= ',F9.4,/,5X,'ENTER 0 TO RETURN TO
     *THE SEARCHER SPRINT/DRIFT SPEEDS MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (S
     *OD,SOS):')
      END
C
C
C    ************************************************************
C    ************************************************************
C
C
C    +----------------------------------------------------------+
```

```
C     |                          SUBROUTINE OPTN12                           |
C     |                                                                      |
C     |                                                                      |
C     +----------------------------------------------------------------------+
C
      SUBROUTINE OPTN12(TD,TS,FILE)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         TD=0.0
         TS=0.0
         GO TO 11
    9 READ (9,*) TD,TS
   11 WRITE(8,100)
      READ(8,*) IOSD
      J=INT(IOSD)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) TD,TS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) TD,TS
      RETURN
   40 CALL CLEAR
      WRITE(8,103)
      READ(8,*) TD,TS
      WRITE(8,102) TD,TS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.12: SEARCHER SPRINT/DRIFT TIMES. ',/,5X,'SEL
     *ECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,
     *'2. ACCEPT EXISTING SPRINT/DRIFT TIMES. ',/,10X,'3. CHANGE SPRINT/
     *DRIFT TIMES. ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OPTION
     * YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
  102 FORMAT(//,'TD= ',F9.4,5X,'TS= ',F9.4,/,5X,'ENTER 0 TO RETURN TO TH
     *E SEARCHER SPRINT/DRIFT TIME MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (T
     *D,TS) (HRS):')
      END
```

```
C
C
C     ********************************************************************
C     ********************************************************************
C
C
C     +------------------------------------------------------------------+
C     |                                                                  |
C     |                        SUBROUTINE OPTN13                         |
C     |                                                                  |
C     +------------------------------------------------------------------+
C
      SUBROUTINE OPTN13(FOMTD,FOMTS,FILE)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         FOMTD=0.0
         FOMTS=0.0
         GO TO 11
    9 READ (9,*) FOMTD,FOMTS
   11 WRITE(8,100)
      READ(8,*) FOMT
      J=INT(FOMT)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) FOMTD,FOMTS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) FOMTD,FOMTS
      RETURN
   40 CALL CLEAR
      WRITE(8,103)
      READ(8,*) FOMTD,FOMTS
      WRITE(8,102) FOMTD,FOMTS
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.13: TARGET FIGURE OF MERIT.      ',/,5X,'SEL
     *ECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,
     *'2. ACCEPT EXISTING FIGURE OF MERIT DATA. ',/,10X,'3. CHANGE FIGUR
     *E OF MERIT DATA. ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OP
```

```
     *TION YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
  102 FORMAT(//,'FOMTD= ',F9.4,5X,'FOMTS= ',F9.4,/,5X,'ENTER 0 TO RETURN
     *TO THE TARGET FIGURE OF MERIT MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (F
     *OMTD,FOMTS):')
      END
C
C
C     ************************************************************
C     ************************************************************
C
C
C     +----------------------------------------------------------+
C     |                                                          |
C     |                    SUBROUTINE OPTN14                     |
C     |                                                          |
C     +----------------------------------------------------------+
C
      SUBROUTINE OPTN14(STMIN,STMAX,FILE)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         STMIN=0.0
         STMAX=0.0
         GO TO 11
    9 READ (9,*) STMIN,STMAX
   11 WRITE(6,100)
      READ(8,*) ST
      J=INT(ST)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(6,101)
      GO TO 11
   20 WRITE(6,102) STMIN,STMAX
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) STMIN,STMAX
      RETURN
   40 CALL CLEAR
      WRITE(6,103)
      READ(8,*) STMIN,STMAX
      WRITE(6,102) STMIN,STMAX
      READ(8,*) NRESET
```

```
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.14: TARGET SPEED RANGE.           ',/,5X,'SEL
     *ECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,
     *'2. ACCEPT EXISTING TARGET SPEED RANGE.    ',/,10X,'3. CHANGE TARGE
     *T SPEED RANGE.    ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OP
     *TION YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
  102 FORMAT(//,'STMIN= ',F9.4,5X,'STMAX= ',F9.4,/,5X,'ENTER C TO RETURN
     *TO THE TARGET SPEED RANGE MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (S
     *TMIN,STMAX):')
      END
C
C
C    ***********************************************************************
C    ***********************************************************************
C
C
C    +---------------------------------------------------------------------+
C    |                                                                     |
C    |                          SUBROUTINE OPTN15                          |
C    |                                                                     |
C    +---------------------------------------------------------------------+
C
C
      SUBROUTINE OPTN15(RTSC,RTCC,FILE)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         RTSC=1.0
         RTCC=1.0
         GO TO 11
    9 READ (9,*) RTSC,RTCC
   11 WRITE(8,100)
      READ(8,*) RTC
      J=INT(RTC)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) RTSC,RTCC
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
```

```
   30 WRITE(7,*) RTSC,RTCC
      RETURN
   40 CALL CLEAR
      WRITE(6,103)
      READ(8,*) RTSC,RTCC
      WRITE(6,102) RTSC,RTCC
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO.15: RATE OF TARGET ZIGS.          ',/,5X,'SEL
     *ECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.',/,10X,
     *'2. ACCEPT EXISTING TARGET ZIG RATES.       ',/,10X,'3. CHANGE TARGE
     *T ZIG RATES.       ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OP
     *TION YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN      *****')
  102 FORMAT(//,'RTSC= ',F9.4,5X,'RTCC= ',F9.4,/,5X,'ENTER 0 TO RETURN T
     *O THE TARGET ZIG RATE MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-PAIR SEPARATED BY A COMMA (R
     *TSC,RTCC)',/,' (RATE OF SPEED CHANGES, RATE OF COURSE CHANGES, PE
     *R HOUR.)')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                      SUBROUTINE OPTN16                         |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN16(ALAM,FILE)
C
      REAL ALAM(3)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         ALAM(1)=1.0
         ALAM(2)=1.0
         ALAM(3)=1.0
         GO TO 11
    9 READ(9,*) (ALAM(I),I=1,3)
   11 WRITE(6,100)
```

```
      READ(8,*) CHLAM
      J=INT(CHLAM)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 12
      GO TO (20,30,40), J
   20 WRITE(8,103) (ALAM(I), I=1,3)
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 CALL CLEAR
         WRITE(7,*) (ALAM(I), I=1,3)
   31 CONTINUE
      RETURN
   40 CALL CLEAR
         WRITE(8,106)
         READ(8,*) (ALAM(I), I=1,3)
   41 CONTINUE
      GO TO 20
   12 WRITE(8,101)
      GO TO 11
C
  100 FORMAT(//,'OPTION NO. 16: ACOUSTIC FLUCTUATION RATE PARAMETERS.',/
     *,5X,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.
     * ',/,10X,'2. ACCEPT EXISTING RATE PARAMETERS.   ',/,10X,'3. CHANGE
     *RATE PARAMETERS.',/,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY SE
     *LECT NO. 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN    *****')
  103 FORMAT(10X,'ALAM(1)= ',F9.4,/,10X,'ALAM(2)= ',F9.4,/,10X,'ALAM(3)=
     * ',F9.4)
  104 FORMAT(/,' ENTER 0 TO RETURN TO ACOUSTIC FLUCTUATION RATE MENU.')
  106 FORMAT(/,' ENTER SEARCHER-LOCAL RATE PARAMETER ALAM(1), GLOBAL RAT
     *E PARAMETER ',/,'ALAM(2), AND TARGET-LOCAL RATE PARAMETER ALAM(3)
     *AS A DATA TRIPLET (ALAM(1),',/,'ALAM(2),ALAM(3)):')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                       SUBROUTINE OPTN17                        |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN17(SIGMA,FILE)
C
```

```
      REAL SIGMA(3)
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         SIGMA(1)=0.0
         SIGMA(2)=0.0
         SIGMA(3)=0.0
         GO TO 11
    9 READ(9,*) (SIGMA(I),I=1,3)
   11 WRITE(8,100)
      READ(8,*) CHSIG
      J=INT(CHSIG)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 12
      GO TO (20,30,40),J
   20 WRITE(8,103) (SIGMA(I), I=1,3)
      WRITE(8,104)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 CALL CLEAR
         WRITE(7,*) (SIGMA(I), I=1,3)
   31 CONTINUE
      RETURN
   40 CALL CLEAR
         WRITE(8,106)
         READ(8,*) (SIGMA(I), I=1,3)
   41 CONTINUE
      GO TO 20
   12 WRITE(8,101)
      GO TO 11
C
  100 FORMAT(//,'OPTION NO. 17: ACOUSTIC FLUCTUATION SCALE PARAMETERS.',
     */,5X,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA
     *.',/,10X,'2. ACCEPT EXISTING SCALE PARAMETERS.    ',/,10X,'3. CHANG
     *E SCALE PARAMETERS.',/,5X,'TO EXIT THIS OPTION YOU MUST EVENTUALLY
     *SELECT NO. 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
  103 FORMAT(10X,'SIGMA(1)= ',F9.4,/,10X,'SIGMA(2)= ',F9.4,/,10X,'SIGMA(
     *2) =',F9.4)
  104 FORMAT(/,' ENTER 0 TO RETURN TO ACOUSTIC FLUCTUATION SCALE MENU.')
  106 FORMAT(/,' ENTER SEARCHER-LOCAL SCALE PARAMETER SIGMA(1), GLOBAL S
     *CALE PARAMETER',/,' SIGMA(2), AND TARGET-LOCAL SCALE PARAMETER SIG
     *MA(3) AS A DATA',/,' TRIPLET (SIGMA(1),SIGMA(2),SIGMA(3)):')
      END
C
C
```

```
C     ************************************************************
C     ************************************************************
C
C
C     +----------------------------------------------------------+
C     |                                                          |
C     |                   SUBROUTINE OPTN18                      |
C     |                                                          |
C     +----------------------------------------------------------+
C
      SUBROUTINE OPTN18(SEED,NREP,TMAX,FILE)
C
      INTEGER SEED
C
      LOGICAL FILE
C
      CALL CLEAR
      IF (FILE) GO TO 9
         SEED=1
         NREP=1
         TMAX=0.0
         GO TO 11
    9 READ (9,*) SEED,NREP,TMAX
   11 WRITE(8,100)
      READ(8,*) RUNS
      J=INT(RUNS)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
   20 WRITE(8,102) SEED,NREP,TMAX
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 WRITE(7,*) SEED,NREP,TMAX
      RETURN
   40 CALL CLEAR
      WRITE(8,103)
      READ(8,*) SEED,NREP,TMAX
      WRITE(8,102) SEED,NREP,TMAX
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
C
  100 FORMAT(//,' OPTION NO. 18: RUN PARAMETERS. ',/,5X,'SELECT ONE OF T
     *HE FOLLOWING:',/,10X,'1. REVIEW EXISTING DATA.
     *',/,10X,'2. ACCEPT EXISTING RUN PARAMETERS. ',/,10X,'3. CHANGE RUN
```

```
     * PARAMETERS. ',/,5X,'SELECT 1, 2, OR 3.',/,5X,'TO EXIT THIS OPTION
     * YOU MUST EVENTUALLY SELECT NUMBER 2.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(//,'SEED= ',I8,5X,'NREP= ',I4,5X,'TMAX= ',F9.4,/,5X,'ENTER
     *C TO RETURN TO THE RUN PARAMETER MENU.')
  103 FORMAT(/,' ENTER NEW VALUES AS A DATA-TRIPLET (SEED,NREP,TMAX):')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                       SUBROUTINE OPTN19                        |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE OPTN19(ECHOS)
C
      LOGICAL ECHOS
C
      CALL CLEAR
   11 WRITE(8,100)
      READ(8,*) ENDE
      J=INT(ENDE)
      IF (J .NE. 1 .AND. J .NE. 2) GO TO 10
      IF (J .EQ. 2) GO TO 12
         ECHOS=.TRUE.
         RETURN
   12 WRITE(8,102)
      READ(8,*) IDNO
      WRITE(6,103) IDNO
      ECHOS=.FALSE.
      RETURN
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
C
  100 FORMAT(/,' OPTION NO. 19: ECHO OF INPUT DATA TO OUTPUT FILE.',/,5X
     *,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. SEND ALL INPUT DATA TO T
     *HE OUTPUT FILE.',/,10X,'2. DO NOT SEND INPUT DATA TO OUTPUT FILE.'
     *//,5X,'IF NO. 2 IS SELECTED, YOU WILL BE ALLOWED TO ASSIGN A RUN I
     *D NUMBER',/,'TO THIS RUN, AND THIS RUN ID WILL BE SENT TO THE OUTP
     *UT FILE.')
  101 FORMAT(/,' ***** INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(/,'ENTER RUN ID NUMBER (UP TO 8 DIGITS, INTEGERS ONLY):')
```

```
  103 FORMAT(/,10X,'RUN IDENTIFICATION NUMBER: ',I8)
      END
C
C
C    ******************************************************************
C    ******************************************************************
C
C
C    +----------------------------------------------------------------+
C    |                                                                |
C    |                       SUBROUTINE OPTN20                        |
C    |                                                                |
C    +----------------------------------------------------------------+
C
      SUBROUTINE OPTN20(STATS1,STATS2)
C
      LOGICAL STATS1,STATS2
C
      CALL CLEAR
   11 WRITE(6,100)
      READ(8,*) SNOS
      J=INT(SNOS)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      IF (J .EQ. 3) STATS1=.FALSE.
      IF (J .EQ. 2) STATS2=.FALSE.
        RETURN
   10 CALL CLEAR
      WRITE(6,101)
      GO TO 11
C
  100 FORMAT(/,' OPTION NO. 20: STATISTICAL ANALYSIS AND OUTPUT.  ',/,5X
     *,'SELECT ONE OF THE FOLLOWING:',/,10X,'1. FULL STATISTICAL ANALYSI
     *S.           ',/,10X,'2. FULL ANALYSIS, BUT OMIT SECTIONING.    '
     */,10X,'3. NO STATISTICAL ANALYSIS (QUICK SUMMARY ONLY).')
  101 FORMAT(/,' *****  INCORRECT RESPONSE, TRY AGAIN     *****')
      END
C
C
C    ******************************************************************
C    ******************************************************************
C
C
C    +----------------------------------------------------------------+
C    |                                                                |
C    |                       SUBROUTINE OPTN0                         |
C    |                                                                |
C    +----------------------------------------------------------------+
C
```

```
      SUBROUTINE OPTNO(GO4IT)
C
      LOGICAL GO4IT
C
      CALL CLEAR
   11 WRITE(8,100)
      READ(8,*) DEF
      J=INT(DEF)
      IF (J .NE. 1 .AND. J .NE. 2 .AND. J .NE. 3) GO TO 10
      GO TO (20,30,40),J
   20 CALL CLEAR
      WRITE(8,102)
      WRITE(8,104)
      WRITE(8,103)
      READ(8,*) NRESET
      CALL CLEAR
      GO TO 11
   30 GO4IT=.TRUE.
      RETURN
   40 GO4IT=.FALSE.
      RETURN
   10 CALL CLEAR
      WRITE(8,101)
      GO TO 11
C
  100 FORMAT(/,' MASTER OPTION: ACCEPT PROGRAM DEFAULTS.',/,5X,'SELECT O
     *NE OF THE FOLLOWING:',/,10X,'1. LIST PROGRAM DEFAULTS.',/,10X,'2.
     *ACCEPT PROGRAM DEFAULTS (START SIMULATION).',/,10X,'3. DO NOT ACCE
     *PT PROGRAM DEFAULTS (COMMENCE OPTIONS).')
  101 FORMAT(/, '***** INCORRECT RESPONSE, TRY AGAIN     *****')
  102 FORMAT(//,' PROGRAM DEFAULTS AS FOLLOWS:',/,5X,'1. A COMPLETE INPU
     *T DATA SET IS ASSUMED TO BE ON A-DISK IN FILE',/,5X,'"PASS DATA" A
     *ND IN PROPER FORMAT FOR DIRECT READING BY PASS.',/,5X,'2. INPUT DA
     *TA IS TO BE SENT TO OUTPUT FILE.',/,5X,'3. ACOUSTIC MODEL IS LAMBD
     *A-SIGMA JUMP.',/,5X,'4. FULL STATISTICAL ANALYSIS IS TO BE DONE.',
     */,5X,'5. SIGNAL INTEGRATION MODEL IS NOT USED.',/,5X,'6. INITIAL T
     *ARGET POSITION IS BIVARIATE UNIFORM (0,XMAX),(0,YMAX).')
  103 FORMAT(//,' ENTER 0 TO RETURN TO DEFAULT OPTION.')
  104 FORMAT(5X,'7. SEARCHER FIGURE OF MERIT IS INDEPENDENT OF TARGET AS
     *PECT.')
      END
C
C
C     **************************************************************
C     **************************************************************
C
C
C     +------------------------------------------------------------+
```

```
C     |                                                                |
C     |                      SUBROUTINE READIT                         |
C     |                                                                |
C     +----------------------------------------------------------------+
C
      SUBROUTINE READIT(XMAX,YMAX,NP,KP,XP,YP,MO,RO,OL,MT,RT,TL,NCZO,
     *RCZO,CZLO,NCZT,RCZT,CZLT,KBAR,START,ANG2,FOMCD,FOMCS,SOC,SOS,TD,TS
     *,FOMTD,FOMTS,STMIN,STMAX,RTSC,RTCC,ALAM,SIGMA,SEED,NREP,TMAX)
C
      REAL XP(50),YP(50),RO(20),OL(20),RT(20),TL(20),RCZO(5,2),CZLO(5),
     *RCZT(5,2),CZLT(5),ALAM(3),SIGMA(3)
C
      INTEGER SEED
C
      READ(9,*) XMAX,YMAX
      WRITE(7,*) XMAX,YMAX
      WRITE(10,*) XMAX, YMAX
C
      READ(9,*) NP,KP
      WRITE(7,*) NP,KP
      WRITE(10,*) NP,KP
      DO 10  I=1,NP
         READ(9,*) XP(I),YP(I)
         WRITE(7,*) XP(I),YP(I)
         WRITE(10,*) XP(I),YP(I)
   10 CONTINUE
C
      READ(9,*) MO,(RO(I),OL(I),  I=1,MO)
      WRITE(7,*) MO
      WRITE(10,*) MO
      DO 11  I=1,MO
         WRITE(7,*) RO(I),OL(I)
         WRITE(10,*) RO(I),OL(I)
   11 CONTINUE
C
      READ(9,*) MT,(RT(I),TL(I),I=1,MT)
      WRITE(7,*) MT
      WRITE(10,*) MT
      DO 12  I=1,MT
         WRITE(7,*) RT(I),TL(I)
         WRITE(10,*) RT(I),TL(I)
   12 CONTINUE
C
      READ(9,*) NCZO
      IF (NCZO .GT. 0) READ(9,*) ((RCZO(I,J),J=1,2),CZLO(I),I=1,NCZO)
      WRITE(7,*) NCZO
      WRITE(10,*) NCZO
      IF (NCZO .EQ. 0) GO TO 13
```

```
      DO 13 I=1,NCZO
         WRITE(7,*) RCZO(I,1),RCZO(I,2),CZLO(I)
         WRITE(10,*) RCZO(I,1),RCZO(I,2),CZLO(I)
   13 CONTINUE
C
      READ(9,*) NCZT
      IF (NCZT .GT. 0) READ(9,*) ((RCZT(I,J),J=1,2),CZLT(I),I=1,NCZT)
      WRITE(7,*) NCZT
      WRITE(10,*) NCZT
      IF (NCZT .EQ. 0) GO TO 14
      DO 14 I=1,NCZT
         WRITE(7,*) RCZT(I,1),RCZT(I,2),CZLT(I)
         WRITE(10,*) RCZT(I,1),RCZT(I,2),CZLT(I)
   14 CONTINUE
C
      READ(9,*) KBAR
      WRITE(7,*) KBAR
      IF (KBAR .EQ. 2) READ(9,*) START,ANG2
      IF (KBAR .EQ. 2) WRITE(7,*) START,ANG2
C
      READ(9,*) FCMOD,FCMOS
      WRITE(7,*) FCMOD,FCMOS
C
      READ(9,*) SOD,SOS
      WRITE(7,*) SOD,SOS
C
      READ(9,*) TD,TS
      WRITE(7,*) TD,TS
C
      READ(9,*) FCMTD,FCMTS
      WRITE(7,*) FCMTD,FCMTS
C
      READ(9,*) STMIN,STMAX
      WRITE(7,*) STMIN,STMAX
C
      READ(9,*) RTSC,RTCC
      WRITE(7,*) RTSC,RTCC
C
      READ(9,*) (ALAM(I),I=1,3)
      WRITE(7,*) (ALAM(I),I=1,3)
C
      READ(9,*) (SIGMA(I),I=1,3)
      WRITE(7,*) (SIGMA(I),I=1,3)
C
      READ(9,*) SEED,NREP,TMAX
      WRITE(7,*) SEED,NREP,TMAX
C
      RETURN
```

```
      END
C
C
C     **************************************************************
C     **************************************************************
C
C
C     +------------------------------------------------------------+
C     |                                                            |
C     |                     SUBROUTINE XDISTB                      |
C     |                                                            |
C     +------------------------------------------------------------+
C
      SUBROUTINE XDISTB(NBINS,PBIN,XBIN,NREP,XT,STACK4)
C
      REAL STACK4(5000),PBIN(20),XBIN(20),SCALE(2),XQ(21),XL(21),XR(21)
C
      INTEGER SEED,C1,C2,C3,C4,KEY(20)
C
      COMMON SEED,C1,C2,C3,C4
C
      C4=C4+1
      IF (C4 .NE. 1) GO TO 10
      SUM=0.0
      NN=NBINS+1
      SCALE(1)=0.0
      SCALE(2)=XBIN(NBINS)
C
      DO 11 I=2,NBINS
         XQ(I)=XBIN(I-1)
         KEY(I)=I
   11 CONTINUE
      XQ(1)=0.0
      XQ(NN)=XBIN(NBINS)
      KEY(1)=1
C
      DO 12 I=1,NN
         XL(I)=XQ(I)
         XR(I)=XQ(I+1)
   12 CONTINUE
C
      CALL SHSORT(PBIN,KEY,NBINS)
C
      DO 13 I=1,NBINS
         SUM=SUM+PBIN(I)
         PBIN(I)=SUM
   13 CONTINUE
C
```

```
      DO 20 K=1,NREP
         J=0
           CALL LRND(SEED,U,1,1,0)
           DO  14 I=1,NBINS
               IF(U .GT. PBIN(I)) GO TO 14
               J=I
               GO TO 21
   14      CONTINUE
   21 CONTINUE
      CALL LRND(SEED,U,1,1,0)
      DELTA=XR(KEY(J))-XL(KEY(J))
      STACK4(K)=XL(KEY(J))+(U*DELTA)
   20 CONTINUE
C
      CALL  FIXP(SCALE)
      CALL  HISTGF(STACK4,NREP,0)
      WRITE(6,100)
      CALL  NOFIXP
C
   10 XT=STACK4(C4)
      RETURN
C
  100 FORMAT(//,5X,'DISTRIBUTION OF INITIAL TARGET X-POSITION',/,10X,'(S
     *PECIFIED BY USER IN BARRIER SCENARIO)')
      END
C
C
C     ******************************************************************
C     ******************************************************************
C
C
C     +----------------------------------------------------------------+
C     |                                                                |
C     |                       SUBROUTINE RELB                          |
C     |                                                                |
C     +----------------------------------------------------------------+
C
C
      SUBROUTINE RELB(TWOPI,THETA,XO,YO,XT,YT,EREL)
C
      REAL PI,PHI
C
      PI=TWOPI/2.0
C
      IF (XT .GE. XO .AND. YT .GE. YO) GO TO 10
      IF (XT .LE. XO .AND. YT .GE. YO) GO TO 11
      IF (XT .LE. XO .AND. YT .LE. YO) GO TO 12
C
```

```
      PHI=((3.0*PI)/2.0)+ATAN((XT-XO)/(YO-YT))
      GO TO 13
C
   10 PHI=ATAN((YT-YO)/(XT-XO))
      GO TO 13
C
   11 PHI=(PI/2.0)+ATAN((XO-XT)/(YT-YO))
      GO TO 13
C
   12 PHI=PI+ATAN((YO-YT)/(XO-XT))
      GO TO 13
C
   13 BREL=THETA+PI-PHI
      IF (BREL .LT. 0.0) BREL=TWOPI-BREL
      BREL=(180.0/PI)*BREL
C
      RETURN
      END
```

APPENDIX I

FORTRAN CODE FOR PASPLT

```
C
C
C     ************************************************************
C     ************************************************************
C     ****                                                    ****
C     ****  PROGRAM "PASPLT" IS A COMPANION PROGRAM TO PROGRAM   ****
C     ****  "PASS" (PASSIVE ACOUSTIC SEARCH SIMULATION). "PASSPLT"****
C     ****  IS THE FORTRAN PROGRAM USED, IN CONJUNCTION WITH THE  ****
C     ****  "DISSPLA" GRAPHICS SYSTEM, TO PRODUCE GRAPHICAL OUTPUT****
C     ****  ON THE TECTRONIX 618 TERMINAL, AND HARD-COPIES AS     ****
C     ****  DESIRED. THE DATA REQUIRED FOR "PASPLT" IS WRITTEN TO ****
C     ****  THE A-DISK (FILE 10) BY "PASS", IN THE PROPER FORMAT  ****
C     ****  FOR USE BY "PASPLT". MOST VARIABLE NAMES USED IN      ****
C     ****  "PASPLT" ARE THE SAME AS USED IN "PASS". ALL SUBROUTINE ****
C     ****  CALLS IN "PASPLT" ARE TO EXTERNAL ROUTINES THAT ARE   ****
C     ****  PART OF THE "DISSPLA" GRAPHICS SYSTEM.                ****
C     ****                                                    ****
C     ************************************************************
C     ************************************************************
C
C
      REAL XP(51),YP(51),RC(20),OL(20),RCZO(5,2),RCZT(5,2),CZLO(5),
     *     CZLT(5),RT(20),TL(20),R(15),C(15),RR(15),CC(15),X(5000),
     *     Y(5000),FO(1000),TO(1000),FT(1000),TT(1000)
C
      LOGICAL ZNDO,ZNDT,AFLLS
C
      ZNDO=.TRUE.
      ZNDT=.TRUE.
      AFLLS=.FALSE.
      READ(10,*) XMAX,YMAX
      READ (10,*) NP,KP
      XX=0.0
      YY=0.0
      DO 10 I=1,NP
         READ(10,*) XP(I),YP(I)
         XX=AMAX1(XX,XP(I))
         YY=AMAX1(YY,YP(I))
 10   CONTINUE
      XMAX=AMAX1(XMAX,XX)
      YMAX=AMAX1(YMAX,YY)
      J=NP+1
      XP(J)=XP(KP)
      YP(J)=YP(KP)
      CALL RESET ('ALL')
      CALL TEK618
      CALL HWSCAL('SCREEN')
      CALL PAGE(11.,8.5)
```

```
      CALL  AREA2D(6.,6.)
      CALL  XNAME('X-POSITION (NM)$',100)
      CALL  YNAME('Y-POSITION (NM)$',100)
      CALL  HEADIN('SEARCHER TRACK$',100,-1.2,1)
      CALL  GRAF(0.0,'SCALE',XMAX,0.0,'SCALE',YMAX)
      CALL  CURVE(XP,YP,J,1)
      CALL  ENDPL(0)
C
      CMAXO=-1.0E+10
      CMINO=1.0E+10
      RMAXO=CMAXO
      RMINO=0.0
      CMAXT=CMAXO
      CMINT=CMINO
      RMAXT=RMAXO
      RMINT=RMINO
      READ(10,*) NO
      DO 11  I=1,NO
         READ(10,*) RO(I),OL(I)
         CMAXO=AMAX1(OL(I),CMAXO)
         CMINO=AMIN1(OL(I),CMINO)
         RMAXO=AMAX1(RO(I),RMAXO)
   11 CONTINUE
      READ(10,*) NT
      DO 12  I=1,NT
         READ(10,*) RT(I),TL(I)
         CMAXT=AMAX1(TL(I),CMAXT)
         CMINT=AMIN1(TL(I),CMINT)
         RMAXT=AMAX1(RT(I),RMAXT)
   12 CONTINUE
      READ(10,*) NCZO
      IF (NCZO .LE. 0) GO TO 16
      DO 13  I=1,NCZO
         READ(10,*) RCZO(I,1),RCZO(I,2),CZLO(I)
         CMAXO=AMAX1(CZLO(I),CMAXO)
         CMINO=AMIN1(CZLO(I),CMINO)
         RMAXO=AMAX1(RCZO(I,2),RMAXO)
   13 CONTINUE
      I=1
      J=1
   20 R(I)=RCZO(J,1)
      C(I)=CMAXO
      R(I+1)=0.5*(RCZO(J,1)+RCZO(J,2))
      C(I+1)=CZLO(J)
      R(I+2)=RCZO(J,2)
      C(I+2)=CMAXO
      I=I+3
      J=J+1
```

```
      IF(J .LE. NCZO) GO TO 20
      K=I-1
   16 READ(10,*) NCZT
      IF (NCZT .LE. 0) GO TO 15
      DO 14 I=1,NCZT
         READ(10,*) RCZT(I,1),RCZT(I,2),CZLT(I)
         CMAXT=AMAX1(CZLT(I),CMAXT)
         CMINT=AMIN1(CZLT(I),CMINT)
         RMAXT=AMAX1(RCZT(I,2),RMAXT)
   14 CONTINUE
      I=1
      J=1
   21 RR(I)=RCZT(J,1)
      CC(I)=CMAXT
      RR(I+1)=0.5*(RCZT(J,1)+RCZT(J,2))
      CC(I+1)=CZLT(J)
      RR(I+2)=RCZC(J,2)
      CC(I+2)=CMAXT
      I=I+3
      J=J+1
      IF (J .LE. NCZT) GO TO 21
      L=I-1
   15 CONTINUE
C
      CALL  RESET ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE(11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('RANGE  (NM)$',100)
      CALL  YNAME('PROPAGATION LOSS  (DB/UPA/YD)$',100)
      CALL  HEADIN('SEARCHER PROPAGATION LOSS  (MODEL)$',100,-1.2,1)
      CALL  GRAF(RMINO,'SCALE',RMAXO,CMAXO,'SCALE',CMINO)
      CALL  RASPLN(2.)
      CALL  CURVE(RO,CL,NC,1)
      IF (NCZO .LE. 0) GO TO 150
      CALL  STEP
      CALL  CURVE(R,C,K,1)
  150 CALL  ENDPL(0)
      CALL  RESET('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE(11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('RANGE  (NM)$',100)
      CALL  YNAME('PROPAGATION LOSS  (DB/UPA/YD)$',100)
      CALL  HEADIN('TARGET PROPAGATION LOSS  (MODEL)$',100,-1.2,1)
      CALL  GRAF(RMINT,'SCALE',RMAXT,CMAXT,'SCALE',CMINT)
```

```
      CALL  RASPLN(2.)
      CALL  CURVE(RT,TL,MT,1)
      IF (NCZT .LE. 0) GO TO 151
      CALL  STEP
      CALL  CURVE(RR,CC,L,1)
  151 CALL  ENDPL(0)
C
      READ(10,*) NPTSO
      NDO=NPTSO-1
      IF (NDO .LT. 10) ZNDO=.FALSE.
      IF (.NOT. ZNDO) GO TO 1000
      READ(10,200) (X(I),Y(I),I=1,NPTSO)
C
      CALL  RESET ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE (11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('TIME (HR)$',100)
      CALL  YNAME('CUMULATIVE PD (UNCONDITIONAL)$',100)
      CALL  HEADIN('CUMULATIVE PROBABILITY OF$',100,-1.2,2)
      CALL  HEADIN('DETECTION VS. TIME$',100,-1.2,2)
      CALL  GRAF(0.0,'SCALE',X(NPTSO),0.0,'SCALE',Y(NPTSO))
      CALL  CURVE(X,Y,NPTSO,0)
      CALL  ENDPL(0)
C
 1000 READ(10,*) NPTST
      NDT=NPTST-1
      IF (NDT .LT. 10) ZNDT=.FALSE.
      IF (.NOT. ZNDT) GO TO 1001
      READ(10,200) (X(I),Y(I),I=1,NPTST)
      CALL  RESET ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE (11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('TIME (HR)$',100)
      CALL  YNAME('CUMULATIVE PCD (UNCONDITIONAL)$',100)
      CALL  HEADIN('CUMULATIVE PROBABILITY OF$',100,-1.2,2)
      CALL  HEADIN('COUNTER-DETECTION VS. TIME$',100,-1.2,2)
      CALL  GRAF(0.0,'SCALE',X(NPTST),0.0,'SCALE',Y(NPTST))
      CALL  CURVE(X,Y,NPTST,0)
      CALL  ENDPL(0)
C
 1001 IF (.NOT. ZNCO) GO TO 1002
      READ(10,200) (X(I),Y(I),I=1,NPTSC)
C
      CALL RESET ('ALL')
```

```
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
      CALL  AREA2C(6.,6.)
      CALL  XNAME('RANGE  (NM)$',100)
      CALL  YNAME('CUMULATIVE PD (CONDITIONAL)$',100)
      CALL  HEADIN('CUMULATIVE PROBABILITY OF$',1C0,-1.2,2)
      CALL  HEADIN('DETECTICN VS. RANGE$',100,-1.2,2)
      CALL  GRAF(C.0,'SCALE',X(NPTSO),0.0,'SCALE',1.C)
      CALL  CLRVE(X,Y,NPTSC,0)
      CALL  ENDPL(C)
C
      READ(1C,20C) (X(I),Y(I),I=1,NDO)
C
      CALL  RESET  ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
      CALL  AREA2C(6.,6.)
      CALL  XNAME('X-POSITICN  (NM)$',100)
      CALL  YNAME('Y-POSITICN  (NM)$',100)
      CALL  HEADIN('SEARCHER PCSITION WHEN$',100,-1.2,2)
      CALL  HEADIN('SEARCHER DETECTS TARGET$',100,-1.2,2)
      CALL  GRAF(C.0,'SCALE',XMAX,0.0,'SCALE',YMAX)
      CALL  CURVE(X,Y,NDC,-1)
      CALL  ENDPL(C)
C
      READ(1C,20C) (X(I),Y(I),I=1,NDO)
C
      CALL  RESET  ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
      CALL  AREA2C(6.,6.)
      CALL  XNAME('X-POSITICN  (NM)$',100)
      CALL  YNAME('Y-POSITICN  (NM)$',100)
      CALL  HEADIN('TARGET POSITION WHEN$',100,-1.2,2)
      CALL  HEADIN('SEARCHER DETECTS TARGET$',100,-1.2,2)
      CALL  GRAF(C.0,'SCALE',XMAX,0.0,'SCALE',YMAX)
      CALL  CURVE(X,Y,NDC,-1)
      CALL  ENDPL(C)
C
 1002 IF (.NCT. INDT) GC TC 1003
      READ(1C,20C) (X(I),Y(I),I=1,NPTST)
      CALL  RESET  ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
```

```
      CALL  AREA2D(6.,6.)
      CALL  XNAME('RANGE  (NM)$',100)
      CALL  YNAME('CUMULATIVE PCD  (CONDITIONAL)$',100)
      CALL  HEADIN('CUMULATIVE  PROBABILITY  OF$',100,-1.2,2)
      CALL  HEADIN('COUNTER-DETECTION  VS.  RANGE$',100,-1.2,2)
      CALL  GRAF(0.0,'SCALE',X(NPTST),0.0,'SCALE',1.0)
      CALL  CURVE(X,Y,NPTST,0)
      CALL  ENDPL(0)
C
      READ(10,200) (X(I),Y(I),I=1,NDT)
C
      CALL  RESET  ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('X-POSITION  (NM)$',100)
      CALL  YNAME('Y-POSITION  (NM)$',100)
      CALL  HEADIN('TARGET  POSITION  WHEN$',100,-1.2,2)
      CALL  HEADIN('TARGET  DETECTS  SEARCHER$',100,-1.2,2)
      CALL  GRAF(0.0,'SCALE',XMAX,0.0,'SCALE',YMAX)
      CALL  CURVE(X,Y,NDT,-1)
      CALL  ENDPL(0)
C
      READ(10,200) (X(I),Y(I),I=1,NDT)
C
      CALL  RESET  ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE  (11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('X-POSITION  (NM)$',100)
      CALL  YNAME('Y-POSITION  (NM)$',100)
      CALL  HEADIN('SEARCHER  POSITION  WHEN$',100,-1.2,2)
      CALL  HEADIN('TARGET  DETECTS  SEARCHER$',100,-1.2,2)
      CALL  GRAF(0.0,'SCALE',XMAX,0.0,'SCALE',YMAX)
      CALL  CURVE(X,Y,NDT,-1)
      CALL  ENDPL(0)
C
 1003 CONTINUE
      READ(10,*) M1
      IF (M1 .LT. 0) AFLLS=.TRUE.
      IF (AFLLS) M1=-M1
      READ(10,200) (FO(I),TO(I),I=1,M1)
      READ(10,*) M2
      READ(10,200) (FT(I),TT(I),I=1,M2)
      YMAX=-10000.0
      YMIN=10000.0
```

```
      XMAX=AMAX1(TO(M1),TT(M2))
      DO 1004 I=1,M1
         YMAX=AMAX1(YMAX,FO(I))
         YMIN=AMIN1(YMIN,FO(I))
 1004 CONTINUE
      DO 1005 I=1,M2
         YMAX=AMAX1(YMAX,FT(I))
         YMIN=AMIN1(YMIN,FT(I))
 1005 CONTINUE
C
      CALL  RESET ('ALL')
      CALL  TEK618
      CALL  HWSCAL('SCREEN')
      CALL  PAGE (11.,8.5)
      CALL  AREA2D(6.,6.)
      CALL  XNAME('TIME$',100)
      CALL  YNAME('FIGURE-OF-MERIT (DB/UPA/YD)$',100)
      CALL  HEADIN('TYPICAL FIGURE-OF-MERIT$',100,-1.2,4)
      IF (AFLLS) CALL HEADIN('MODEL: LAMBDA-SIGMA JUMP$',100,-1.2,4)
      IF (.NOT. AFLLS) CALL HEADIN('MODEL: GAUSS-MARKOV$',100,-1.2,4)
      CALL  HEADIN('SOLID LINE: SEARCHER$',100,-1.2,4)
      CALL  HEADIN('DOTTED LINE: TARGET$',100,-1.2,4)
      CALL  GRAF(0.0,'SCALE',XMAX,YMIN,'SCALE',YMAX)
      IF (AFLLS) CALL STEP
      CALL  CURVE(TO,FO,M1,0)
      CALL  DOT
      CALL  CURVE(TT,FT,M2,0)
      CALL  ENDPL(0)
C
      CALL  DONEPL
      STOP
C
  200 FORMAT(8F9.3)
      END
```

LIST OF REFERENCES

1. Koopman, B. O., Search Theory and Screening, Pergamon Press, 1980.

2. Washburn, A. R., Search and Detection, Military Applications Section of the Operations Research Society of America, 1981.

3. Naval Postgraduate School NPS55-82-020, The Optimal Search for a Moving Target When the Search Path is Constrained, by J. N. Eagle, 1982.

4. Brown, S. S., "Optimal Search for a Moving Target in Discrete Time and Space", Operations Research, Vol. 28, 1275-1289, 1980.

5. Stewart, T. J., "Experience With a Branch and Bound Algorithm for Constrained Searcher Motion" in Search Theory and Applications, K. B. Haley and L. D. Stone, editors. Plenum Press , 1980.

6. Stone, L. D. and Kadane, J. B., "Optimal Whereabouts Search for a Moving Target", Operations Research, Vol 29, 1154-1166, 1981.

7. Washburn, A. R., "On a Search for a Moving Target", Naval Research Logistics Quarterly, Vol. 27, 315-321, 1980.

8. Daniel H. Wagner Associates, memorandum report to Commander Submarine Development Group Two, Documentation of a Monte-Carlo Simulation of Submarine ASW Search, by L. Arnold, 10 June 1975.

9. Applied Mathematics Inc., technical report, User's Guide: Program Search, by W. J. Browning, 1982.

10. Daniel H. Wagner Associates Report to the Office of Naval Research (Contrac N00014-73-0432), A Comparison of Detection Models Used in ASW Operations Analysis, by McCabe, B. J. and Belkin, B., 1973.

11. Center for Naval Analysis, Report CRC395, An Introduction to the Modeling of Cumulative Probability of Detection in Underwater Acoustics, by W. J. Hurley, 1980.

12. Tehan, T. N., A Survey of Parameter Estimation Methods for the Lambda-Sigma Jump Detection Model, M. S. Thesis, Naval Postgraduate School, June, 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Libary, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Professor James N. Eagle Code 55ER Naval Postgraduate School Monterey, California 93943	2
4.	Professor R. N. Forrest Code 55FO Naval Postgraduate School Monterey, California 93943	1
5.	Professor D. P. Gaver Code 55GV Naval Postgraduate School Monterey, California 93943	1
6.	Commander Submarine Development Squadron Twelve U.S. Naval Submarine Base Groton, Connecticut 06340	1
7.	LCDR Steven G. Slaton 3 Pleasant Street Westerly, Rhode Island 02891	1

NDERY
AS

207693

Thesis
S570243 Slaton
c.1
A computer program to model passive acoustic antisubmarine search using Monte Carlo simulation Techniques.

AMERICAN BINDERY
TOPEKA KANSAS

thesS570243
A computer program to model passive acou
3 2768 001 00611 7
DUDLEY KNOX LIBRARY